Adaptive music: Automated music composition and distribution

Technical Report 2016

David D. Albarracín Molina

Advisor: Francisco J. Vico, Ph.D.

Departamento de. Lenguajes y Ciencias de la Computación

Universidad de Málaga, Spain

2016

Abstract

Creativity, or the ability to produce new useful ideas, is commonly associated to the human being, but there are many other examples in nature where this phenomenon can be observed. Inspired by this fact, in engineering and particularly in computational sciences, many different models have been developed to tackle a number of problems.

Music, a form of art broadly present along the human history, is the main field addressed in this thesis, taking advantage of the kind of ideas that bring diversity and creativity to nature and computation. We present Melomics, an algorithmic composition method based on evolutionary search, with a genetic encoding of the solutions that are interpreted in a complex developmental process that leads to music in standard formats.

This bioinspired compositional system has exhibited a highly creative power and versatility to produce music of different type, which in many occasions has proven to be indistinguishable from the music made by human composers. The system also has enabled the emergence of a set of completely novel applications: from effective tools that help anyone to easily obtain the precise music they need, to radically new uses like adaptive music for therapy, amusement or many other purposes. It is clear to us that there is yet much research to be done in this field and that countless and new unimaginable uses will derive from it.

Acknowledgements

I have had the privilege to work in a laboratory provided by Universidad de Málaga in the PTA, such a beautiful environment, and to have been funded by the spin-off program of this university and by the national programs INNPACTO and Plan Avanza2, essential mechanisms in this country, to enhance the so necessary technological and research initiatives.

Thanks to Francisco Vico, being part of the GEB, a surprisingly heterogeneous place has allowed me to work along with many singular and interesting people, who all have influenced in me and I consider them all my friends. Jose, a natural inquirer, was the first contact that I had with music theory since high school, and then came Gustavo, dealing with music from a more formal perspective, to whom I also thanks his patience, especially at the beginning of this my re-learning. Thanks to Carlos, we helped each other since the beginning, during those nights of writing proposals; and Fran Moreno, since those nights of coding scripts. Thanks to Juanca and Guti for sharing their wisdom and for those discussions about both relevant and insubstantial matters; Miguel, the ultimate sysadmin and programmer, also with a charming personality; Jose David, a wide and probably endless source of relevant information, who also showed me, together with Dani and Gema how beautiful a research environment like this could be. I do not forget Julio, providing his experienced help in technical situations and his peculiar sense of humor; Curro, definitely helpful in any bureaucratic process; and many others with whom I have shared very interesting thoughts and established a good friendship: Ellen, Vicente, Rafa, Mario, Enrique, Héctor, Cristóbal, Pablo, Pedro, Fran Palma, Oliver, Juan Arenas, Juan Travesedo, Sirenia and Chelo. I also remember people outside the group that I have had the opportunity to work with, like Cory McKay, always disposed to work on the defined lines and to help with MIR matters; the people from the department of psychology of the UGR and the people from the neonatal unit at the hospital Montecanal.

I especially thank my parents Ana María and Andrés José, whose true support and open minded education have undoubtedly allowed me to choose to make this humble contribution. Thanks to my brother Andrés and my sister Mariam, who inevitably have to bear me and learn about my work. To my grandparents and the rest of my family, for their nice company and support.

Thank you all.

Credits

This thesis is largely a consequence of a research project commonly known as Melomics, driven by the GEB[1], Universidad de Málaga and mostly funded and promoted by two national research projects granted in 2010 and 2011, and a university spin-off award in 2010. Many people have been involved in one way of another. Below we give credit to the people whose work has directly made this thesis possible:

- The present compositional system, starting from a collection of scripts and ideas, provided by my advisor Francisco Vico, has continuously been improved with the addition of countless functionalities in the synthesis, score-writing and core systems, which have been coded with the help of Francisco Moreno Arcas, Rafael De Vicente Milans, Pedro Guillén Aroca and José David Gutiérrez Gutiérrez; and specified with the help of the musical experts Gustavo Díaz Jerez, Juan Carlos Moya Olea, José F. Serrano García, Óliver Moya Bueno and Francisco J. Palma Olmos.
- To design and build the mobile applications and web tools, essential parts of the adaptive music system described in Chapter 4, we counted on: Miguel Carmona Bermúdez, Cristóbal Carnero Liñán, Héctor Mesa Jiménez, Pablo Moreno Olalla, Salvador Burrezo González, José David Gutiérrez Gutiérrez, Enrique Morris Cordón and Mario A. Ureña Martínez.
- The perceptive study described in Section 3.3.1, based on an unpublished work designed, executed and written in collaboration with Francisco Vico, Alfredo Raglio, Francisco Rivas-Ruiz, Juan C. Moya Olea and Carlos A. Sánchez Quintana.

[1] www.geb.uma.es

Contents

List of figures

List of tables

List of musical content

Chapter 1

Introduction

Specifying what is music and distinguishing it from other kind of sounds is a complex issue and depends on many cultural factors, which is why, the definitions are usually expressed in a broad form, like the one given by the Collins Dictionary:[2] "any sequence of sounds perceived as pleasing or harmonious". Music compositions, as a form of art, do influence the audience; they raise on it some kind of emotions or psychological effects (Juslin & Sloboda, 2001), sometimes in an intended way by the composer. As, we may think, the composition process, the notation methods, the instruments, the styles, etc. have evolved since their emergence in the prehistoric eras, always linked to the social progression and the technological advancements (Crocker, 1966) (Meyer, 1989) (Christensen, 2002) (Sachs, 2012). Nowadays music is taking advantage from the current digital means, such as new musical instruments or computer programs. These tools ease and improve the processes of composing and performing music, also leading to the appearance of new genres and, especially, new ways of making music (Fernandez & Vico, 2013).

In this thesis we present a computational system aimed to create original full-scale musical compositions. The methods that we will describe take advantage of some models in current computational intelligence, in order to achieve the intended creativity on its creations. Furthermore, some innovative applications with automatically generated music will be shown.

[2] http://www.collinsdictionary.com/

Biomimetics is a branch of the engineering, consisting of mimicking solutions that are observed in nature, either structures or behaviors. Over the years this approach has shown its potential to solve complex problems in computer science. Some of the most known and relevant methods are artificial neural networks, fuzzy logic, evolutionary algorithms or ant colony optimization (Gendreau & Potvin, 2010). Biomimetic algorithms are regularly used because they can bring very creative and almost unpredictable solutions. These results are not always optimal, but they can fit the problem reasonably well, besides, sometimes the problem is not supposed to comprise a unique optimal solution. Hence, we will describe how we use this less common way of computing as the basis for the proposed music composition system.

A tool capable of producing pieces of music automatically, with similar characteristics to those created by humans, can bring a number of advantages, for instance: being able to generate a huge quantity of music in a short period of time; or having the possibility, for anyone, to find or produce the kind of music they want, through some few directions; or making easy to explore new musical genres or styles; or being able to produce families of compositions, a group of themes holding some parameters in common and being different in some others. This thesis explores some of these new paths arisen from automated composition.

Before attending the rest of the topics of this dissertation, some relevant concepts are introduced: In Section 1.1 we focus on what we understand as *computational creativity*, and propose some case-studies. In Section 1.2 we expose how the music field in general is benefiting from computers, and we describe different existing methods for composing in an automated way. In Section 1.3, we outline how computer science do take advantage by mimicking nature. Finally, Section 1.4 summarizes the work presented in the dissertation.

1.1 Computational creativity

Creativity is an idea still difficult to define. In fact, the current concept has been forged during the modern eras, the Renaissance, the Enlightenment and so on; in previous ages, especially the middle ages, what we understand now as creating something, was seen more like a manner of discovery, imitation, or even a divine inspiration; Gods could have this ability, but not humans (Runco & Albert, 2010) (Tatarkiewicz, 1980). Then, the creativity was started to be perceived as a capability of great men, in contrast to one just being productive (Dacey, 1999); and it was clear, as it is now, that it is linked to the capability of imagination (Tatarkiewicz, 1980). There is a multitude of definitions, but a suitable one to us can be creativity as *the ability to generate novel and valuable ideas* (Boden, 2009).

Creativity has generally been seen as a characteristic of the human being, distinguishing them from the rest of beings, in the same way it happen with the intelligence; so, it stands to reason that these two qualities may be related (Sternberg & O'Hara, 1999). Being a distinctive feature might carry certain skepticism when attributed to other entities, but building machines that exhibit this capability certainly represents a challenge for scientist and engineers. From the point of view of computer science, this is a cutting-edge field in computational intelligence, usually addressing complex and vague problems, with ill-defined spaces of search (Gero & Maher, 1993) (De Smedt, 2013), so that using conventional algorithms is not feasible here.

As we have suggested, creativity is about combining pre-existing ideas or things to produce a new and valuable one. Taking the model given by (Wallas, 1926), we can summarize the process in four stages: (1) gathering inputs, like aims, pre-existing solutions, limitations, etc.; (2) exploring ideas, by combining the initial products, trying new approaches, etc.; (3) generate effective results and (4) testing the results. Hence, it seems a process that can be modeled into a machine, at least in the case of specific purposes.

Along the development of computer science, artificial creativity has regularly been approached in arts, the main area associated to this ability. For example there are programs that can make original paintings (Cook & Colton, 2011), videos (Chambel, et al., 2007) or writings (Gervás, 2009) (Turner, 2014). But creativity has also been treated in some other fields, for example in AI to play board games having a large space of search. In the game GO, strategies based on more flexible algorithms than the ones used in chess, for example, are used (Müller M. , 2002), and in Selfo (Vico F. J., 2007), strategies based on self-organization observed in certain living organisms are applied (Albarracín-

Molina, 2010). Another field where creativity is explored is in AI for computer games; nowadays we can find, for example, cooperation between group of non-player characters, like sharing of tasks, or AI players exhibiting unpredicted reactions to different situations, such as new visual information, events in the environment, perceived sounds, human player presence or other non-player character actions (Yannakakis, 2012). A specific case of creativity is given by the computer called Watson, a system by IBM which includes natural language processing, information retrieval, knowledge representation, automated reasoning and machine learning technologies (Ferrucci, et al., 2010). The system proved its capability of producing original output by competing and defeating the human players in the TV game show Jeopardy! (Markoff, 2011). Since then, Watson has been used for applications involving understanding questions asked by humans and providing valuable answers, including purposes as diverse as financial applications, medical problems, legal consultory or even its use in cooking (Ferrucci, Levas, Bagchi, Gondek, & Mueller, 2013).

In research there always has been an interest about creativity, it has been studied in nature (Bentley P. J., 1999) or in computer science (Colton, López de Mantaras, & Stock, 2009), it has been discussed how to measure it (Ritchie, 2007) (Bringsjord, Bello, & Ferrucci, 2003) (Pease & Colton, 2011) or how to distinguish creative systems from automated systems that just pretend to be creative (Colton, 2008); and in computational intelligence, many approaches have been followed to achieve creativity, for example with evolutionary computation being one of the most frequently used methodologies (Bentley & Corne, 2002) (Boden, 2009).

The system that we propose incorporates techniques mainly from three areas of computational intelligence: (1) Knowledge-based systems, which can be implemented in the form of rules for a specific domain, and have been used before in computer music, for example by (Schaffer & McGee, 1997). We make use of a knowledge-based approach to build the general rules for composing (e.g. physical limitations of musical instruments) or for specifying musical styles (see Section 2.2.3.1), where the pursued novelty is achieved by a guided exploration of the parameters. (2) Formal grammars, which can be roughly defined as a vocabulary of symbols plus a set of rewriting rules that guide the development of an initial character into some possible strings of symbols. These methods have been used for example in computational design (Stiny, 1980), (Shea & Fenves, 1997). In our system we use two indirect encoding models based on grammars for representing musical genomes, which would be developed into legible compositions. (3) Search methods, which includes a multitude of algorithms of different nature in order to explore different spaces of search, in many cases including the use of heuristics (Russell & Norvig, 2009) (Campbell, Hoane, & Hsu, 2002). We have implemented an evolutionary search based on assessing the compositions after being developed from the genomes; if they do not fit certain general or style-based musical requirements, they can be discarded, but if they do, they can be used to feed a defined genetic pool. A number of case-studies will be described to show the potential of these methodologies to produce creative results. Furthermore, the genetic-based encoding proposed may help us to build new search methods, enabling for example the possibility of implementing a more conventional genetic algorithm (see Section 5.2.1).

There is evidence to suggest that the output from Melomics system is actually creative: Its scores have been interpreted by top-shelf musicians, including the London Symphony Orchestra (Rao, 2012); frequently people, including professional musicians, mistakenly attribute Melomics works to human styles and even to specific composers; some of its compositions have been selected to compile musical albums and to be premiered in live concerts[3] [4] [5] (Ball, 2012); and large collections of themes have been acquired by third parties to different uses. We have also had to handle several issues triggered by the fact of an autonomous system producing genuine creative outcomes, as the case occurred in December 2014. On that occasion, Melomics entries in Wikipedia[6] experienced a massive deletion of its media contents due to a copyright-related controversy. These contents were after restored, since it was settled that there was "no creative input from a human".

[3] https://www.youtube.com/watch?v=bD7l4Kg1Rt8
[4] https://www.youtube.com/watch?v=JOksICT8DZU
[5] https://www.youtube.com/watch?v=PzrcoqpnZqA
[6] https://commons.wikimedia.org/w/index.php?title=Commons:Village_pump/Copyright&oldid=143770481#melomics.com

1.2 Computer music and algorithmic composition

Music, as every other field in human knowledge, has benefited from the appearance and development of computers and computer science. New tools to help in the process of creating music have appear: recording and reproducing sounds, mixing and editing audio, writing scores, synthesizing; and even digital audio workstations (DAW) have been developed, integrating most of these functionalities. There is also a kind of tool known as Computer-Aided Algorithmic Composition (CAAC) for helping in the composition process, such as Csound (Boulanger, 2000), SuperCollider (McCartney, 2002) or MAX/MSP (Puckette, 2002). New musical instruments have been invented, like the electric violin (Washington, DC: U.S. Patent and Trademark Office Patent No. D403,013, 1998); virtual instrument technologies (Rossum & Joint, 1995) (Steinberg, 1999); and standards for storing music, either in a symbolic way (MIDI,[7] SIB,[8] XML[9]) or as a wave (WAV,[10] FLAC,[11] MP3[12]). The digital era has brought yet another step in the evolution of musical genres (Pachet & Cazaly, 2000) and the development of algorithmic composition, a discipline at the intersection of music theory and computational intelligence.

Along with the invention of transistor computers in 1950s, the earliest works of algorithmic composition aroused, roughly at the same time as the concept of artificial intelligence, although the two fields did not converge until sometime later. As an overview, some works in formal composition at the early age were: an unpublished work by Caplin and Prinz in 1955 (Ariza, 2011), with an implementation of Mozart's dice dame and a generator of melodic lines using stochastic transitional probabilities; in 1956, the pioneering work Hiller and Isaacson's Illiac Suite (Hiller Jr & Isaacson, 1957), based on Markov chains and rule systems; MUSICOMP by Baker (Ames, 1987), implementing some methods used by Hillers; Xenakis's stochastic algorithms in early 1960s, working as CAAC (Ames, 1987); in 1961, a dedicated computer was able to compose new

[7] http://www.midi.org/
[8] http://www.sibelius.com/
[9] http://www.musicxml.com/
[10] http://www-mmsp.ece.mcgill.ca/Documents/AudioFormats/WAVE/WAVE.html
[11] https://xiph.org/flac/
[12] http://www.iis.fraunhofer.de/en/ff/amm/prod/audiocodec/audiocodecs/mp3.html

melodies related to previous ones, by using Markov processes (Olson & Belar, 1961); in 1963, Gill's algorithm applied classical AI techniques (hierarchical search with backtracking) (Gill, 1963); PROJECT1, created by Koenig in 1964 (Ames, 1987), used serial composition and other techniques; and also in 1964 Padberg implemented a compositional framework based on procedural techniques (Padberg, 1964). Machines were becoming less expensive and more powerful, so algorithmic composition field has been always growing, however, it is being a slow process, with difficult communication between scientists and artists, most of the initiatives being carried out by the latter and with little continuity in research.

Algorithmic composition aims the automation of tasks in the creative process, such as the definition of melodies, rhythms and harmonies, or the introduction of counterpoints, voice leading, articulations or effects. It can be approached from mimicking a specific style from a set of musical examples, or by modeling the composer's methodology. In both cases we obtain tools to support musicians in their creative work, or fully autonomous systems capable of producing complete compositions.

Composing music, treated as an imitation problem, has been solved with relatively high success, see for example (Cope, 1992) or (Pachet, 2002). On the other hand, composing with no creative input, has been a harder problem for computational intelligence; partly because of the difficulty of assessing the results, although some frameworks have been proposed (Gero J. S., 2000) (Pearce & Wiggins, 2001) (Ritchie, 2007) (Boden, 2009). The algorithmic techniques in the compositional systems observed in the literature can be classified in four general categories: (a) Symbolic AI, whose main representatives in algorithmic composition are grammars and rule-based systems. They have been used both for imitating musical styles or specific composers and for automating the composition process. These are usually very labor-intensive methods, since they require the musical knowledge to be encoded and maintained in the symbolic framework. (b) Machine learning, comprising Markov chains and artificial neural networks, whose nature have made them generally more suitable for imitation purposes. (c) Optimization techniques, having been used for imitation and for the automation of compositional tasks, and counting on the evolutionary algorithms as the most representative example. (d) Complex-systems like self-similarity and cellular automata, used to generate novel material without being restricted to the human knowledge. For a deeper review see (Fernandez & Vico, 2013).

As shown, there have been a number of attempts to build automatic composition systems, using many diverse approaches. However, most of them seems to be scattered works, with only a few of them having achieved mainstream awareness or widespread use, like for example CHORAL (Ebcioglu, 1988), EMI (Cope, 1992) or Melomics (Sánchez, Moreno, Albarracin, Fernandez, & Vico, 2013), the latter developed in this dissertation. Melomics has recently attracted a great media attention (Europa-Press, 2012) (Peckham, 2013); it has been used to populate a huge web repository of music of different genres,[13] with two original albums premiered so far; and it has served as a source of creative material for professional composers (Diaz-Jerez, 2011) and as the basis for innovative applications, like adaptive music, which will be explained in the later chapters. It seems clear that, eventually, automatic composition will completely change the way the music is produced and distributed; and soon, composing will be at reach of anyone, even for people with no background in music theory.

[13] www.geb.uma.es/melomics/melomics.html

1.3 Biological inspiration

As it is fairly known, nature is a powerful source of inspiration to solve many different kinds of problems. In engineering and sciences, this approach called biomimetics, has made possible great inventions to appear, such as, the well-known Velcro, copying the tiny hooks on the surface of burs; a multitude of designs of functional structures, like the wings of an airplane, the diving fins or some climbing tools that mimic the anatomy of geckos; the radar and sonar; self-cleaning or self-repairing surfaces; and countless other examples. Particularly, in computer science, biomimetics plays a core role in the field of computational intelligence, being artificial neural networks, fuzzy logic, evolutionary algorithms or swarm intelligence, some of its most representative methods, being applied to solve a multitude of computational problems, such as financial trade, computer vision, networks traffic, thoroughfare traffic, control systems in appliances, creative industrial design or arts.

Evolution is one of the main mechanisms that have contributed to the diversity and complexity in nature. These mechanisms have inspired in computer science evolutionary algorithms, which have been applied in many diverse domains. This methodology is based on having an initial population of solutions which undergoes a cyclic process of reproduction (with possible variation), evaluation and selection. Classical evolutionary algorithms use direct encoding, this is, an explicit map between representations and solutions (genotypes and phenotypes). This approach has traditionally suffered from some problems (Hornby & Pollack, 2002), like difficult scalability due to the size of the solutions and the search spaces; or unstructured solutions, for example different parts of the genotype growing uncoordinatedly. In biology, the process that transforms a zygote into a full multicellular organism, with the cells dividing and arranging themselves forming tissues and complex shapes, a development process regulated by the selective expression of genes (Carroll, Grenier, & Weatherbee, 2004) (Forgács & Newman, 2005) (Mayr, 1961), is known to play a key role in the evolutionary processes (Marcus, 2003) (Lewontin, 2000) (Müller G. B., 2007) (Bateson, 2001) (Borenstein & Krakauer, 2008). Evolutionary developmental biology (known as evo-devo) (Carroll S. B., 2005) is the branch studying these mechanisms, which consider that the evolution of diversity is related to the evolution of the developmental process (Borenstein & Krakauer, 2008). In fact, there is evidence that the sole variation in the regulatory programs may better explain the evolutionary emergence of novelty and diversity (Levine & Tjian, 2003), (Davidson, 2010), (Hood & Galas, 2003), (Davidson & Erwin, 2006). In computer science, this school has inspired analogous methods. With evolutionary algorithms, if using an adequate indirect encoding (Stanley & Miikkulainen, 2003), a small genotype can result in a large and complex phenotype, making the approach much more scalable. Moreover, small variations in the genotype could produce greatly different phenotypes, since the changes would be propagated in a structured way during the developmental process, which would cause this method to produce more robust, structured and potentially original solutions. Hence, these methods have been used in fields that demanded exploring a huge space of options or certain creativity, for example, as a tool for variating existing solutions toward desired targets or as an automated tool for brainstorming, which have traditionally been carried out by humans. Some real-life examples are the design of NASA satellites antennas (Hornby, Lohn, & Linden, 2011), the design of microstructured optical fibers (Manos, Large, & Poladian, 2007), the automatic generation of board games (Browne, 2008) or new techniques for automatic character animation (Lobo, Fernández, & Vico, 2012).

In biology, the embryological development is not an isolated process solely directed by the genome, it is also studied as a growth influenced by the physical nature of the environment (Forgács & Newman, 2005). From this point of view, to understand the emergence of diversity, physical interactions are as important as genes in the developmental processes, and they can have a great impact in the evolutionary dynamics in some contexts like early stages of multicellular evolution (Newman, 2003) or in some unicellular organisms (Goodwin, 2001). In computer science these ideas have also inspired a number of works, such as (Lobo D. , 2010), (Fernández, 2012) or (Sánchez-Quintana, 2014). Melomics is another example; the development of a genome into a fully understandable composition is conditioned by the provided musical context, as will be detailed in further sections.

As summary, Melomics takes two main ideas from biomimetic discipline: an indirect genetic representation for compositions and a complex developmental process affected by a soft of physical context, which we identify with the music style. This approach enables us to exploit a set of advantages (as will be detailed in Chapter 2), among others, Melomics has a great potential to produce original material with no creative input; it makes possible to obtain useful variations from a given solution; it is able to explore a huge space of search, providing a great variability in the solutions; and it is a robust and scalable system, it can be used to explore new regions of the space.

1.4 Outline

The rest of this thesis is organized as follows: Chapter 2 introduces Melomics, a computer system capable of producing human-like music, both in score and audio formats. It makes use of evolutionary computation, specifically two models of genetic representations and their associated development procedures that give place to the actual compositions. In Chapter 3, the most relevant tools for studying musical aspects are presented. They study music from different perspectives, treating symbolic notation, wave formats and cultural data. These tools will be used to analyze the properties of Melomics music and compare it with the traditional human-made music. This chapter also described some experimental studies with Melomics, from a more subjective approach. Chapter 4 describes how the automated method is exploited to create a new way of composing and providing with music, that is able to adapt it in real time, according to a given context; then some actual applications will be detailed. In Chapter 5 the main results, conclusions and contributions of this work are presented and discussed, as well as some extended lines of work, Appendix A gives a description of each symbol of the representation models, Appendix B shows the main hardware used for developing and testing, Appendix C provides some execution statistics for both the atonal and the tonal system, using different hardware and configuration of parameters, and Appendix D consists of an extensive summary and conclusions of this thesis, in Spanish.

Chapter 2

An algorithmic composition method

In this chapter, we present the computational system called Melomics, which applies bioinspired approaches to compose music. The software includes two different genetic-based ways of representations for music compositions, in both cases mimicking a complex developmental process inspired by biological ontogeny, after which the music pieces appear written in a more explicit notation. There are also implemented different mechanisms at some stages of the process, in order to assess the goodness of each composition and feed the system. A fully developed composition, stored in the system in its particular format, can then be prepared and converted to many standard file types, either musical score formats (MusicXML, MIDI) or audio formats (FLAC, MP3).

This chapter is organized as follows: In Section 2.1 we introduce the basis and motivation to use the chosen approach and we put it in context with some other existing methods. In Section 2.2 we detail the two formal models which constitute the foundations of our compositional algorithms; we present some implementation aspects and we give some relevant use-cases for both systems. In Section 2.3 we describe our internal representation of the music and how it is translated into the preferred output formats. In Section 2.4 we provide an overview of the whole process and we discuss our goals and results.

2.1 Motivation

Automatic music composition is a challenging problem from both the technical and the scientific perspective. From a technical point of view, one important matter is the design of the musical formats, which should allow representing and managing the information in a suitable way, according to the concrete purposes. For example, a it could be required to support rapid and flexible basic operations like raising the musical pitch to one single channel, or changing the tempo of the whole composition; but also more advanced manipulations, like dynamic adjustment of the composition responding to certain changes, such as adjusting the tessitura and the melodic steps of one track, when its performing instrument has been changed. There are many other technical concerns that must be tackled in automated composition, for example, the management of the directions for the musical performance, which are usually included in the scores by the authors, as expressions in natural language. The system must be able to produce and handle this kind of instructions and to incorporate it into the different musical format, with the less possible loss of information. From a more scientific perspective, as it has been pointed out, music is able to produce a psychological influence in the audience, which can be represented, for example, with the valence-arousal model (Kim, Lee, Kim, & Yoo, 2011). Dealing with this kind of problems would require, on the one hand, a mechanism to describe music in a higher level of abstraction than usual (our first proposal is commented in 2.2.3.1 Musical styles) and, on the other hand, it is needed a method capable of exploring and finding diverse, valuable and novel results in the huge and highly unbounded space of search, where the solutions are a priori unknown.

One of the main properties of our genetics models is that they are based on formal grammars. A composition is described using a set of rules that expand high-level symbols into a more detailed sequence. Since hierarchical structures can be identified in most kind of musical pieces, this approach has been very commonly used, particularly through regular and context-free grammars (Fernandez & Vico, 2013). Lindenmayer Systems (abbreviated L-systems) (Lindenmayer, 1968), a particular variant that served us to implement the first genetic model, works by performing a parallel rewriting of all the applicable rules in each iteration. They are characterized by the capability to encode relatively complex structures with simple sets of rules, and they have been especially used to model the anatomy, physiology and morphogenesis of plants (Prusinkiewicz & Lindenmayer, 1990). Implementing a version of the so called turtle interpretation, using musical elements instead of graphical information, has brought many applications to algorithmic composition, such as (Prusinkiewicz, 1986), (Mason & Saffle, 1994) or (Wilson, 2009). Using grammars as a genomic basis in a genetic algorithm for automated composition, has also been tried, from different approaches, like the system described by (Reddin, McDermott, & O'Neill, 2009) or GeNotator (Thywissen, 1999), with an interactive fitness function. Another property of our model, as already indicated, is that it is strongly driven by knowledge-based functions. Including some kind of composition rules at some point in the workflow is very common in algorithmic composition systems, some examples are (Thomas, 1985), which performed four-part chorale harmonization, (Thomas, Chatterjee, & Maimone, 1989) for simple melody generation, (Steels, 1979) for finding passing chords, (Horowitz, 1995) for making improvised jazz solos or (Gjerdingen, 1988) for generation of counterpoints.

2.2 Representation models

We have developed two different compositional methods, one aimed to produce atonal music, characterized by the lack of strong rules for tonal arrangement, and the second is intended to create tonal compositions, prioritizing harmony rules, this is, dealing with specific relationships between note pitches along time and the participant instruments. We have design a particular genetic representation for each system, both based in formal grammar. The two workflows slightly vary mainly due to their different genetic formats and the different ways to evaluate the goodness of the results. However, apart from the harmony, the treatment of the rest of the musical variables is very similar in both type of music: certain type of order is sought in the compositional structure, in the durations (rhythms), in the volumes (dynamics) or the timber (ensemble of instruments); conventions in music notation or formats are equal; and, from a general point of view, what is expected from both system is the same as well, solutions that are original, valuable, mutable, etc. So the systems have been designed following similar principles.

2.2.1 Music variables

In the design of the representation models, two of the main problems that have been addressed are (a) how to map the structure of the genetic representation with the structure of a composition and (b) how to assign the operators and the rest of the genes to music behaviors and parameters, thus converting an arbitrary string of symbols into musical data.

- **The structure** of a composition can be described by the arrangement of certain identifiable sections or musical units. Some of these entities can be at the higher level of the musical hierarchy, such as the verses, the choruses or the bridges; and some other at the lower levels, like for example musical motifs or specific patterns. These units can be usually recognized for their particular musical features (rhythm, melodic contour, tone, performing instruments...). A complete composition shall be formed by some of these pieces of original material, which are reused along time, possibly introducing variations. In our system, the musical structure will be handled by the grammar structure, providing high compactness, readability, scalability, expressivity and flexibility.
- **The instruments** will define the timbres or tone colors in the composition, but they will also shape the musical functions or behaviors to be performed. These roles can be of many types and may define the relationship among the participant instruments. Some examples are melodies, counterpoints, harmonizations, harmonic accompaniments, rhythmic accompaniments or FX. In our system, the roles and the concrete instruments for a composition can be specified through predesigned templates, and then handled by a set of rules based on expert knowledge, which will influence the process of creating genetic material. Regarding the audio file that might be asked to be produced, the actual virtual instruments to use during the synthesis are also managed by the expert knowledge and coded in the genotype.

- **The textural evolution** is referred to the presence or absence of each instrument in the composition over time. The associated parameters will settle instrumental densities, persistent musical roles or transitions among them during the composition. This feature is handled at the higher level of the grammar, where the proper regulatory genes, mostly introduced by expert knowledge rules, are in charge of enabling or inhibit the presence of roles or instruments.

- **The tempo** marks the pace of the composition, usually expressed as the number of beats (a note value of reference) per units of time (e.g. 60 beats per minute). As in the musical compositions, in the genetic representation, the tempo can be changed at any place, but it is typically set globally or in the highest level of the grammar structure. There is an alternate or complementary way of specifying the tempo, based on subjective directions given by the composer (e.g. allegro, andante, molto vivace...), which is especially used in classical music. Our system will handle these expressions, through some simple rules, at the final stage of development, when building the score or during the synthesis process.

- **The measure** (or bar) is a segment of time containing a certain number of beats, indicated by the measure number, normally implying some kind of grouping among these beats, conditioning the resulting rhythm. For example, in a section with a measure number 3/4, the rhythmic patterns that are expected to emerge will consist of a sequence of 3 beats, each one filled with a note or a cluster of notes of equivalent duration. Additionally, the measure number may influence the internal accentuation or the way the notes are performed in terms of intensity according to their position in the subdivision of one beat.

- **The rhythm** refers to a sequence of notes and rests with specific durations, which can appear repeatedly over time. The kind of sequences might be influenced by the chosen measures and many other variables like the role who is executing the rhythm, the actual chosen instrument to play the role, the musical genre or style, etc. In most of western music, the rhythm is built using a duration of reference and a set of note values representing durations obtained by multiplying the reference by a power of two (positive or negative). All these note values can be used as it, tied among them or be split or grouped into some other more irregular configurations. This method makes all kind of sequences of durations possible, but more frequent those which are simpler. In the genetic models we have tried to map this behavior with the corresponding structural genes.

- **The pitch** in music is the property of sound related to the frequency of vibration, letting us to distinguish bass from treble sounds. Music harmony consists of the way of using the different pitches simultaneously and along the time. It takes into account the proportions between the fundamental frequencies of the involved notes, because it will affect the way the sounds are perceived inside the listener's auditory system. For example the ratio 11/12 is perceived as tense or dissonant, while 4/3, 3/2, 2/1 or 1/1 will be more pleasant. In music composition and, as consequence, in the proposed systems, managing harmony has a lot of implications. For example in the lowest level, the genetic representation include structural genes that helps to represent the pitches as shifts in the chosen scale (explained next) and also to build chords, arrangements of notes with specific ratios among their pitches. In the higher levels, tools based on expert knowledge have been designed to consider harmony, especially in the system for tonal music. There are methods to manage chord progressions, build or settle the musical scales, introduce different kind of harmonic accompaniments or locate and filter certain kind of dissonances once the current composition has been built. One last consideration regarding the pitch property is about its bounds. Actual instruments usually have upper and lower pitch limits, commonly known as tessitura, beyond which is not possible to produce a sound. Additionally, the instruments usually have what is called sweet spot, a region of pitches where they make the clearest sounds. In some other cases, these limits may be imposed by the style of music that is being created.

- **The scale** is a parameter related to pitch and harmony. In this compositional system, as usually in music composition, managing pitches is done in an abstract way; shifts among pitches (simultaneous or over time) are considered in relation to a concrete mode, or collection of allowed pitches, plus an initial tone of reference called key (also root or tonic) giving place to an scale, as a result. The mode can be settled globally or can be changed among different sections of a composition. The keynote usually changes more often, producing different harmonic progressions, which causes for example that a musical fragment repeated with a shift on the root note, could be identified by the listener as a variation of the same musical material. In our model for atonal music, handling the scale is a relatively flexible and dynamic process, while the model for tonal music counts on a much more controlled way to do so.

- **The dynamics** are related to the way the notes are produced with the instruments in terms of strength, this is, the resulting volume but also its progression over time. This parameter can be studied at different levels; in the higher there are the dynamic directions to indicate the musicians how the general volume should be, for example "*mf*" for *mezzo-forte* or moderately loud, "*pp*" for *pianissimo* or very soft and "*cresc.*" (also represented with a special symbol) for an increasing volume. In the lower levels, attack and decay times of notes are considered; musical articulations tell the musician how to play certain notes, for example *legato* is used to indicate that notes should be tied while performing them and *staccato* to indicate short attacks and decays. At this level, it is also considered the internal accentuation, namely, the way the subdivisions of each beat are accentuated. It depends on some parameters like the current measure number, the chosen rhythm or the given genre or style. We have designed structural genes to express concrete dynamics or modulations at any levels, although they are typically used at the higher levels to configure the global curve or to make an instrument come in or come out with a fade effect. There are also structural genes to express the articulations and to specify internal accentuations.
- **The effects** can be of many different types. We have introduced genes to implement most of the musical effect that can be found in music composition, some of these operators are designed to provide with directions to the musicians on the score (e.g. the *pizzicato* direction) and

some others to be used during the process of synthesis (e.g. the reverb effect). There are some effects that are typically aimed to influence the whole composition (e.g. volume normalization), others to affect a particular track or instrument (e.g. panning) and a third class to be applied over one note or a group of them (e.g. *glissando* or *flutter-tonguing*).

2.2.2 Model for atonal music

To build the first compositional method we considered and picked some of the element present in the modern western music theory. In general, this was done to favor the sequences of sounds and rests to appear organized, which is somehow what we can understand as music. These elements include note values, to enhance order in rhythm; key and scales, to enable order in pitches; or structural components like measures, phrases, etc., to promote order in a higher level of a musical composition. However, we avoided to use specific constructions or to follow the usual conventions of western music, such as rhythmic patterns or musical modes and chords, in order to open the range of musical constructions that would be produced.

- Sample 2.1. Demo of themes with a wide configuration of parameters.

Since the basic ingredients for generating music were implemented but no specific conventions were hardcoded in the genetic model, the system was suitable to aim to the contemporary classical music, an important area in the nowadays professional composition. This style comprehends several kinds of artistic tendencies present in the late 20th century, with a special mention to the modernism, and it is mainly characterized for the lack of strong rules to manage the musical elements, giving the author the freedom to explore combinations of the available variables. This is specially noted in the pitch dimension, generally producing music in the chromatic scale with little restrictions, thus being perceived with no defined key or tonal center, which is called atonality.

- Sample 2.2. Example of atonal music produced by Melomics.

The core of the atonal system is based on a grammar model, which holds the primitive genetic representation of a composition, and a developmental process, which consist of unfolding the grammar, and interpreting the resulting string of symbols into musical elements. In the different stages of this process, there are implemented complementary functions, applied with parameters stored outside the genotype, which constitutes a complementary mechanism that helps to give the final shape to the resulting composition.

In the atonal system we have also designed a first input interface of high level musical parameters that can help anyone, even with no knowledge of the software, to specify the type of music to be produced. Some of these features, on the one hand, are aimed to guide the process of building genomes, making them more probable to develop into songs that fit these specifications. Examples of this kind of parameters are textural density curve, repetitiveness of pitches or repetitiveness of rhythms. Although we use indirect encoding, the design of the genetic representation allows us to execute certain types of interventions like the ones mentioned, with relatively little effort, thus making faster the whole process of search. On the other hand, there are musical features like amount of dissonance or shape of melodic curves, which shall be applied to assess the resulting music, letting to reach the end of the process only to those pieces that satisfy all the specifications, discarding the others. In any case, this information will feed the system by saving or discarding parts of the genetic material (the right side of the production rules), to be used in the next executions.

In the next sections we will describe the formal basis of our model for atonal music: (1) the genetic representation of songs and (2) the process to develop a genome into an explicit composition. Then we will present some use-case experiments carried out with this compositional system, serving as a timeline to illustrate the introduction of some new musical operator, a new relevant functionality or tool or a particular and interesting use of this technology.

2.2.2.1 Model of genome

The genetic model in the atonal system can be described as a deterministic L-system plus some parameters to settle the iterations during the rewriting process and some others to give specific values when translating the resulting string into music. These later will be a duration of reference (used in several occasions during the whole process of composition and transcription to output formats), the scale for each track, the instrument ID and its tessitura. The L-system is defined by the triplet:

$G = (V, S, P)$

where

V is the alphabet, a finite set of symbols,

S is the axiom or starting symbol,

P is the set of production rules with only one symbol on the left side, which will drive a parallel rewriting process from the axiom to the final string. Being a deterministic system means that for each symbol, there is only one rule with it on the left side. In our implementation, if no rule is written for a symbol A, the rule $A \rightarrow A$ is assumed.

The symbols with no rule specified for them, will act later as operators in charge of managing the musical parameters (pitch, duration, onset time, effects, volume, tempo…) during the translation of the final string into music, as we will show in the next section. On the other hand, symbols with an explicit production rule having them on the left side, will represent either structural units of the composition or specific musical notes, depending on how the grammar iterates; in any case, they will have an instrument ID associated.

Below we introduce some of the reserved symbols (music operators) used in the system and their respective meaning in musical terms:

$1 increases pitch value one step in the scale.

$2 decreases pitch value one step in the scale.

$5 saves current pitch and duration values in the stack PS.

$6 returns to the last value of pitch and duration stored in the stack PS.

$7 saves current time position in the stack TS.

$8 returns to the last time position saved in TS.

$96 applies dynamic mezzo-forte.

@60$102 applies the tempo: quarter equal 60.

For an extensive explanation of the reserved symbols see Appendix A.1.

Next we give the definition for G_a, a simple example illustrating an instantiation of this model:

$G_a = (V, S, P)$

$V = \{\#0, \#1, \#2, \#3, \#4, \$1, \$2, \$5, \$6, \$7, \$8, \$96, @60\$102\}$

$S = \{\#0\}$

and P consisting of the following rules:

$\#0 \rightarrow @60\$102\$96\#2\$1\$1\$1\$1\$1\$1\$1\#2$

$\#1 \rightarrow \#1$

$\#2 \rightarrow \$7\#3\$8\#4$

$\#3 \rightarrow \#3\$2\#3\$1\$1\#1\#3$

$\#4 \rightarrow \#4\#1\$5\$1\$1\$1\#4\$6\#4$

The final meaning of the structure and symbols will depend on how the grammar is developed, as will be explained in the next section. Nevertheless, looking at the example, we can bring forward some conclusions. For example, the production rule associated to the axiom, symbol #0, would represent the higher level of the composition, the structure will consist of two identical musical units, represented by symbol #2, but the second one being shifted eight times in the pitch dimension (which is the meaning of the symbol $1), implying that the second half of the composition will sound in a similar way but with a higher pitch. Another conclusion is that on each period there will be probably two instruments performing notes, represented by the genes #3 and #4, unless the grammar iterates only once, in whose case there will be just one instrument, the associated to symbol #2, or if it does not iterate at all, in whose case the only note being played will be #0, the axiom. We usually use the symbol #1 to represent the musical rest, by assigning it the proper instrument ID, although it is not a reserved symbol.

Generally the structure will be built using templates designed according to expert knowledge, as previously commented. The right side of the production rules are built using material form the genetic pool, which is filled with that belonging to compositions that were scored high by the assessment functions in previous generations. These sequences of symbols are randomly altered (deleting, adding or modifying symbols) and then adjusted to assure that they satisfy as possible the current specifications, before being added to the new genome.

One of the main advantages of this encoding method is its robustness, the symbols can be altered or the rules be reorganized; and the genome is still valid and readable. In fact the compositions coming from a genome and a slight variation of it are commonly very similar to each other, for example they can have the same structure with different melodies or vice versa, they can be the same composition but with different rhythms, tone or set of instruments, etc. The recursive self-similarity provided by the L-system model, favoring to produce similar structures at the different levels of a composition, also resulted useful as a way of producing order in some of the addressed music styles.

2.2.2.2 Model of development

In the process of obtaining the phenotype (the music explicitly written in any on the considered output formats) starting from the genotype (a text file with the extension ".gen" containing the grammar and parameters), the first stage is to produce the resulting string of symbols, after a rewriting process driven by the L-system. Since in this kind of grammar there is no distinction between terminal and non-terminal symbols, we introduced a simple function to stop applying each production rule. Each of them will have an associated value (r_i) indicating the possibility to be applied in the next rewriting iteration. There is a global parameter (T), which is the initial value for all r_i and there is a particular parameter for each production rule (I_i) to compute the new value for r_i, after the rule i has been applied in the current iteration, with the formula given below:

$$r_i = max(0, round(r_i \cdot I_i - 1))$$

Thus when $r_i = 0$, the rule will not be used anymore.

Using the example introduced in the previous section, we will illustrate a rewriting process, with the r_i current values shown at the left of each production rule and the current resulting string at the bottom; with $T = 1$ and $I_i = 1, \forall i$.

Iteration 0

Remaining iterations	Rules
1	#0 → @60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2
1	#1 → #1
1	#2 → \$7#3\$8#4
1	#3 → #3\$2#3\$1\$1#1#3
1	#4 → #4#1\$5\$1\$1\$1#4\$6#4
Resulting string	#0

Iteration 1

Remaining iterations	Rules
0	#0 → @60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2
1	#1 → #1
1	#2 → \$7#3\$8#4
1	#3 → #3\$2#3\$1\$1#1#3
1	#4 → #4#1\$5\$1\$1\$1#4\$6#4
Resulting string	@60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2

Iteration 2

Remaining iterations	Rules
0	#0 → @60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2
1	#1 → #1
0	#2 → \$7#3\$8#4

1	#3 → #3\$2#3\$1\$1#1#3
1	#4 → #4#1\$5\$1\$1\$1#4\$6#4
Resulting string	@60\$102\$96\$7#3\$8#4\$1\$1\$1\$1\$1\$1\$1\$7#3\$8#4

Iteration 3

Remaining iterations	Rules
0	#0 → @60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2
1	#1 → #1
0	#2 → \$7#3\$8#4
0	#3 → #3\$2#3\$1\$1#1#3
0	#4 → #4#1\$5\$1\$1\$1#4\$6#4
Resulting string	@60\$102\$96 \$7#3\$2#3\$1\$1#1#3\$8#4#1\$5\$1\$1\$1#4\$6#4 \$1\$1\$1\$1\$1\$1\$1 \$7#3\$2#3\$1\$1#1#3\$8#4#1\$5\$1\$1\$1#4\$6#4

Iteration 4

Remaining iterations	Rules
0	#0 → @60\$102\$96#2\$1\$1\$1\$1\$1\$1\$1#2
0	#1 → #1
0	#2 → \$7#3\$8#4
0	#3 → #3\$2#3\$1\$1#1#3
0	#4 → #4#1\$5\$1\$1\$1#4\$6#4
Resulting string	@60\$102\$96 \$7#3\$2#3\$1\$1#1#3\$8#4#1\$5\$1\$1\$1#4\$6#4 \$1\$1\$1\$1\$1\$1\$1 \$7#3\$2#3\$1\$1#1#3\$8#4#1\$5\$1\$1\$1#4\$6#4

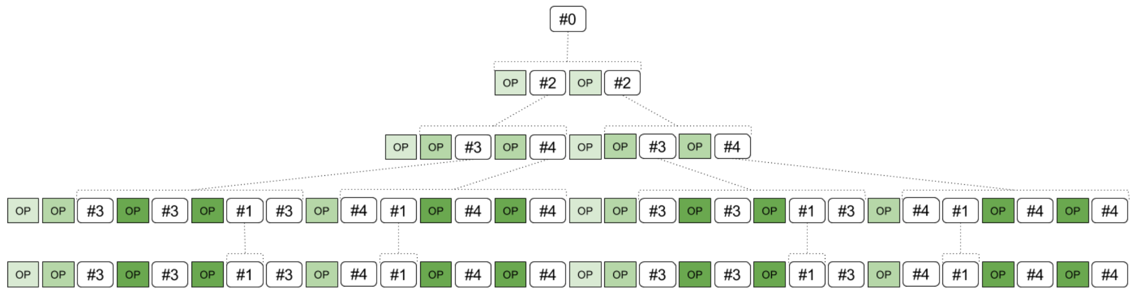


Figure 2.1. Example of atonal music. Developmental tree. The operators have been summarized as "OP".

The string produced at the end of this process will suffer additional adjustments. Some of these operations are aimed to clean the string from useless genes before the translation into musical elements, for example deleting the consecutive repetition of an idempotent music operator. Some other readjustments are provoked by the physical context, for example depending on the chosen instruments, it may be necessary to rewrite some genes to satisfy certain constraints, like maximum number of consecutive notes or the emergence or suppression of certain musical effects.

Translating the final string to musical elements, the second phase of the developmental process, is an implementation in musical terms of the turtle method, consisting of a sequential reading of the string, from left to right, where the general behavior proceeds as follows: Each time a gene representing a music operator appears, the value of the corresponding musical variable is shifted, starting from an initial value, established by default, or indicated in the current genome. Other operators just set a specific value for a musical variable, and a third kind is used to store certain values in a stack, in order to make them available in the future. When a gene representing a note is read, the present value of each musical parameter is applied to the musical note being produced.

With the previous example, we will illustrate how would be the resulting composition if the string is interpreted on each iteration (in the actual execution, this is only done after the last iteration). As initial values required, we will use the C major scale, a quarter note as default duration, a tempo equal 80, a default dynamic mezzo-piano, the initial pitch for all tracks MIDI-60 and the following instruments to be assigned to the production rules: piano, rest, church organ, church organ and cello.

Iteration 0

String: #0

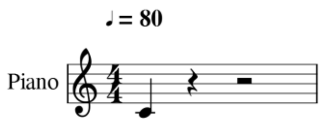


Figure 2.2. Example of atonal music. Resulting score at iteration 0.

- Sample 2.3. Atonal example. Iteration 0.

With no iterations, there is only the axiom symbol as performing instrument in the resulting string, which produces a single note played by its associated instrument, the piano, with all the musical parameters on their default values.

Iteration 1

String: @60$102$96#2$1$1$1$1$1$1$1#2

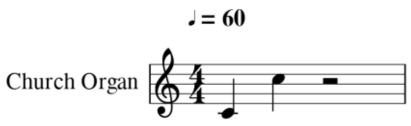


Figure 2.3. Example of atonal music. Resulting score at iteration 1.

- Sample 2.4. Atonal example. Iteration 1.

After one iteration from the axiom, the composition has new values for tempo and dynamics, and two notes with a shift of 7 steps in the pitch. Because we are using the C major scale, the second note will be played one octave higher. The instrument performing this simple sort of a melody will be the one associated to symbol #2, the church organ.

Iteration 2

String: @60$102$96$7#3$8#4$1$1$1$1$1$1$1$7#3$8#4

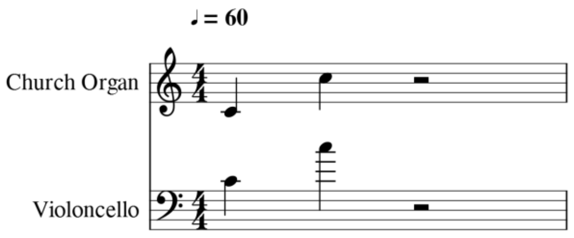


Figure 2.4. Example of atonal music. Resulting score at iteration 2.

- Sample 2.5. Atonal example. Iteration 2.

The development has passed the second level of the genomic structure, giving place to the two final instruments, church organ and cello, playing simultaneously the same notes.

Iterations 3 and 4

String:

@60$102$96$7#3$2#3$1$1#1#3$8#4#1$5$1$1$1#4$6#4$1$1$1$1$1$1$1

$7#3$2#3$1$1#1#3$8#4#1$5$1$1$1#4$6#4

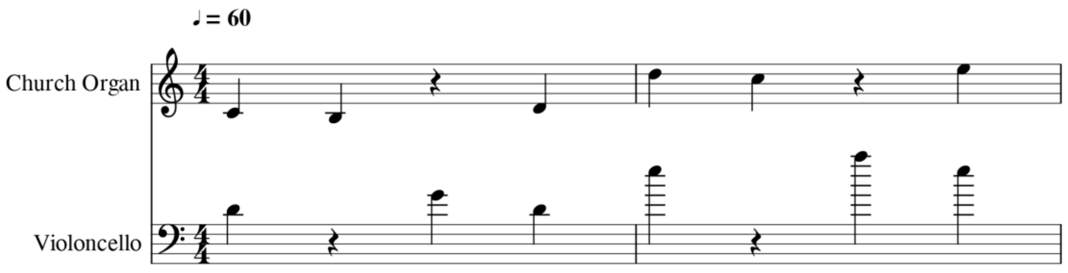


Figure 2.5. Example of atonal music. Resulting score at iteration 4.

- Sample 2.6. Atonal example. Iteration 4.

At this level, the genes #0 and #2 do not represent any actual notes, otherwise they can be understood as higher level compositional entities. We can interpret #0 as the composition and #2 as the two phrases. The rules for the symbols #3 and #4 have been developed once and they have produced two different melodies, slightly more complex than before, and including some rest notes (symbol #1). If there would be an additional iteration of these rules, the genes

#3 and #4 will act as both high-level compositional entities and as instruments performing the notes.

The musical content will be stored in an internal structure, described in Section 2.3.1, which contains meta-information of the compositional process and the composition itself, including global properties and values for each track and note in an explicit form, similar to the way it is arranged in a MIDI file. This representation has been designed both to be easy to manipulate and to be easy to transform into the different output formats, including the printed score, the editable score and the audio file. Before being presented in one of these final forms, the musical information in the internal structure undergoes an additional series of adjustments, as occurred in the previous representation. The changes are, as well, some of them to put the musical content in a more manageable form and some others provoked by the physical context: constraints in tessituras, discretization of note durations or concrete implementations of musical effects.

2.2.2.3 Relevant experiments

Iamus Opus #1

After building the core of the system and during a long period of debugging, we were obtaining some promising preliminary results, thus we got the attention of some experts in music. Particularly we started collaborating with Gustavo Díaz-Jerez,[14] a Spanish pianist, composer and professor in Centro Superior de Música del País Vasco "Musikene" and Centro Superior de Música Katarina Gurska, who will later describe in a paper part of his work in this project (Diaz-Jerez, 2011).

Iamus Opus One, written for flute, clarinet, horn, violin and violoncello, was created on October 15, 2010 and it is the first piece of professional contemporary classical music composed by a computer in its own style, and it was the result of the first directed search using the algorithm. It was necessary to introduce certain constraints in the system, in order to produce music that could be performed with real instruments; some of these boundaries were maximum allowed polyphony, tessituras, possible effects, allowed note

[14] https://en.wikipedia.org/wiki/Gustavo_D%C3%ADaz-Jerez

durations and dynamics, etc. Then we introduced some new functions to assess the compositions after being developed; they measured the textural evolution, the amount of dissonance and the repetition of note-values and note-pitches. The method and the thresholds to discriminate the wellness of a composition were fine-tuned in an iterative fashion with the help of the expert. The general goal pursued was to obtain pieces of music with a limited amount of perceived dissonance and whose fundamental musical materials (specific melodic constructions, structure…) were evoked along the time and among the different participant instruments.

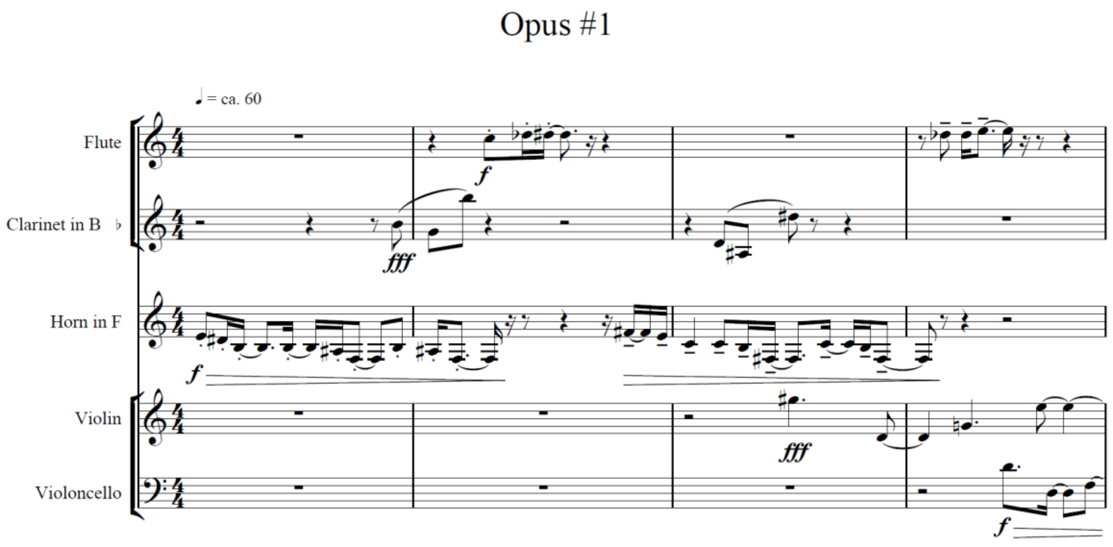


Figure 2.6. Excerpt of Iamus Opus #1.

- Sample 2.7. Iamus Opus #1 full score.
- Sample 2.8. Iamus Opus #1 audio emulation.

Hello World!

One year after Opus #1 was created, we worked on four main aspects of the system: (1) the fitness function of the evolutionary system was refined; (2) the capability to compose for real instruments was increased, handling more types of instruments, more performing techniques and the music content produced for each track, being more suitable to the instrument in charge of performing it; (3) the functions to write the output MusicXML file were enhanced, so the produced score was richer and able to show most of the elements of the standard music notation; and (4) the process of building genomes was divided in two phases, one controlling the structure and another to build the low level material (phrases and motives).

The system was tested with a particular configuration of parameters and a specific ensemble of instruments. The computer cluster Iamus[15] (see Appendix B.2) was programmed to run Melomics software for about ten hours, creating independent compositions for clarinet, violin and piano. At the end, one composition was arbitrary picked among the bundle of hundreds that were produced, which was called Hello World!, making reference to the computer program Hello World!.[16] This piece of contemporary classical music was composed in September 2011 and premiered on October 15, 2011 at the Keroxen music festival in Santa Cruz de Tenerife, Spain (EL DÍA, 2011) (Redacción Creativa, 2011), being arguably the first full-scale work entirely composed by a computer and using conventional music notation.

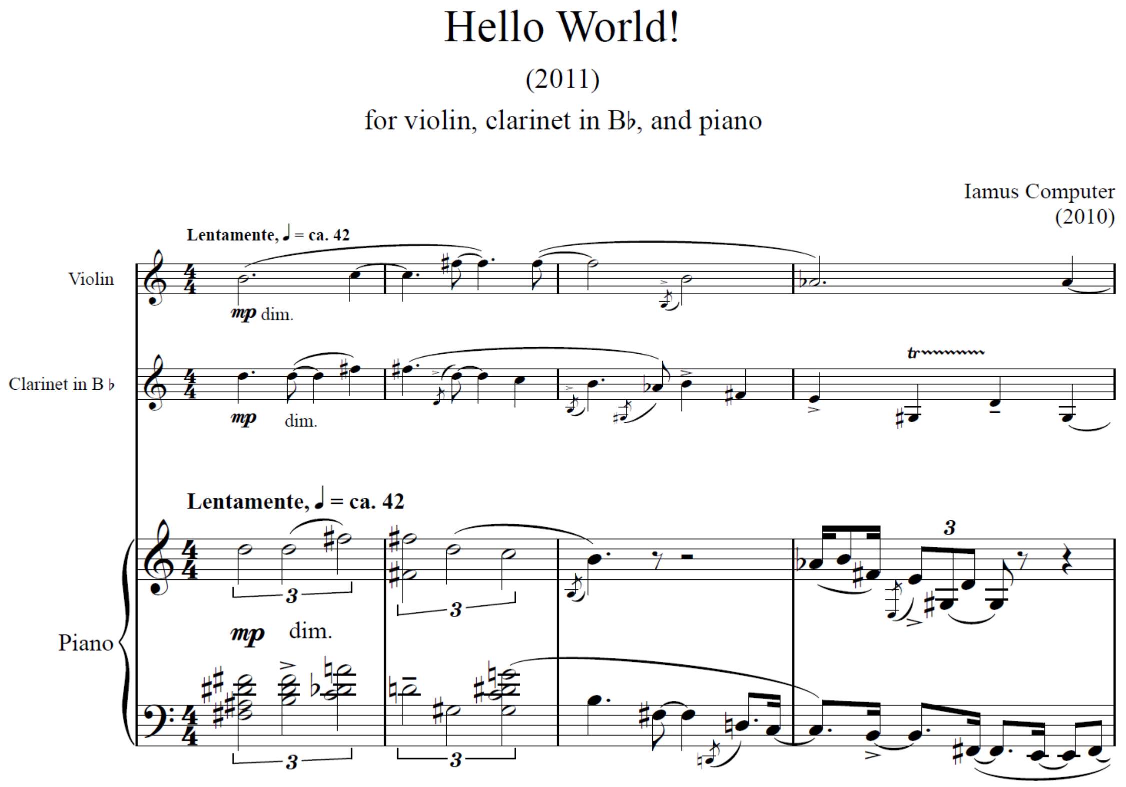


Figure 2.7. Excerpt of Hello World!.

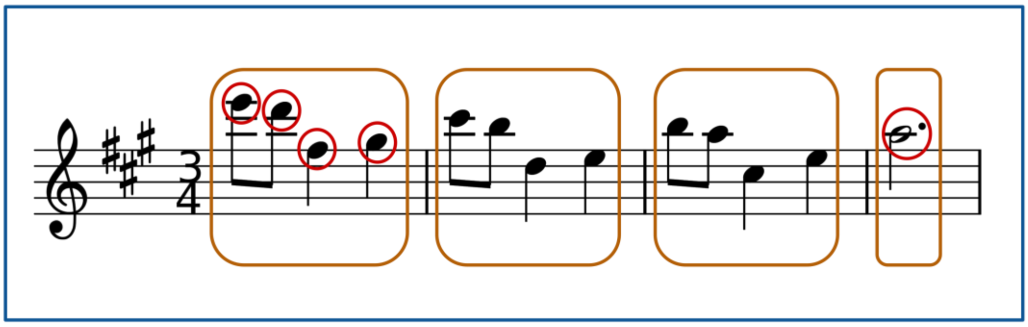
[15] https://en.wikipedia.org/wiki/Iamus_(computer)
[16] https://en.wikipedia.org/wiki/%22Hello,_World!%22_program

- Sample 2.9. Hello World! full score.
- Sample 2.10. Audio of Hello World! premiered in Tenerife.

Genetic engineering: Nokia tune

Using a recognizable example, we tested one of the main benefits from having a genetic representation for the compositions, that is, a high power to produce coherent variations. We implemented some tools to perform this genetic engineering: a crossover mechanism, which takes global parameters, track properties and production rules from either of the two parent genomes, giving place to an offspring with inherited musical features; and some primary mutation functions, to alter the values of the musical variables at the different structural levels; allowing to intervene over singular notes, the relationships among them, the relationships between instruments, changing instruments, scales, structure, instrumental effects, performance techniques or global parameters (like tempo or dynamics). We choose the Nokia tune™[17] for being a simple and very identifiable theme. Since one musical composition can theoretically have infinite representations in our genomic model, we carried out a well-studied process of reverse engineering to bring the piece into a genome that not only would develop into the desired theme, but it also encode the structure supposedly followed when it was created. Then we run a set of mutations and crossovers with some random themes from the system, producing as a result a bundle of themes that can be considered as relatives of the original tune.

[17] https://en.wikipedia.org/wiki/Nokia_tune



$\#0 \longrightarrow \boxed{\mathbf{\#2}}$

$\#1 \longrightarrow \mathbf{\#1}$

$\#2 \longrightarrow \boxed{\mathbf{\#3}}\ \$2\$2\ \boxed{\mathbf{\#3}}\ \$2\ \boxed{\mathbf{\#3}}\ \$2\ \boxed{\mathbf{\#4}}$

$\#3 \longrightarrow \$4\ \enclose{circle}{\mathbf{\#5}}\ \$2\ \enclose{circle}{\mathbf{\#5}}\ \$3\$2\$2\$2\$2\$2\ \enclose{circle}{\mathbf{\#5}}\ \$1\ \enclose{circle}{\mathbf{\#5}}$

$\#4 \longrightarrow \$3\$3\$3\$3\ \enclose{circle}{\mathbf{\#5}}$

$\#5 \longrightarrow \mathbf{\#5}$

Figure 2.8. Reverse engineering experiment. The tune was reengineer using three compositional levels: Phrase, idea and notes. For clarity, we have omitted the global parameters, the track parameters and some of the music operators in the representation of the genome.

- Sample 2.11. Audio of Nokia tune reengineered.
- Sample 2.12. Audio of Nokia tune relative 1.
- Sample 2.13. Audio of Nokia tune relative 2.
- Sample 2.14. Audio of Nokia tune relative 3.
- Sample 2.15. Audio of Nokia tune relative 4.

Themes for relaxing applications

Besides the targeted contemporary classical style, Melomics software was used to provide with music in a research project with the same name. It was aimed to create a method based in music therapy with biofeedback to reduce anxiety. The system, which was previously designed to produce full compositions, was configured for this other target; with the help of experts in music therapy, we configured the functions to produce simpler samples. In general, it was focused on a different and more restricted space of search.

- Sample 2.16. Audio of theme for relaxing applications. Example 1.
- Sample 2.17. Audio of theme for relaxing applications. Example 2.

At this time we made the first test of producing a genetic family of themes that could be used together as one singular music entity in this kind of applications. Each version of the theme can be selected during the real-time therapy, according to certain criteria; this behavior being the basis of a more sophisticated method designed for adaptive applications that will be introduced in Chapter 4.

- Sample 2.18. Evolving audio for relaxing applications.

Web repository and Iamus album

The version of the system used for creating Hello World! was refined and a higher level user interface was implemented, with a graphic tool allowing to describe the expected evolution of some parameters along the duration of the composition. These improvements made easier to describe compositional styles, establishing the basis for the future tool used in the tonal system, described in Section 2.2.3.1.

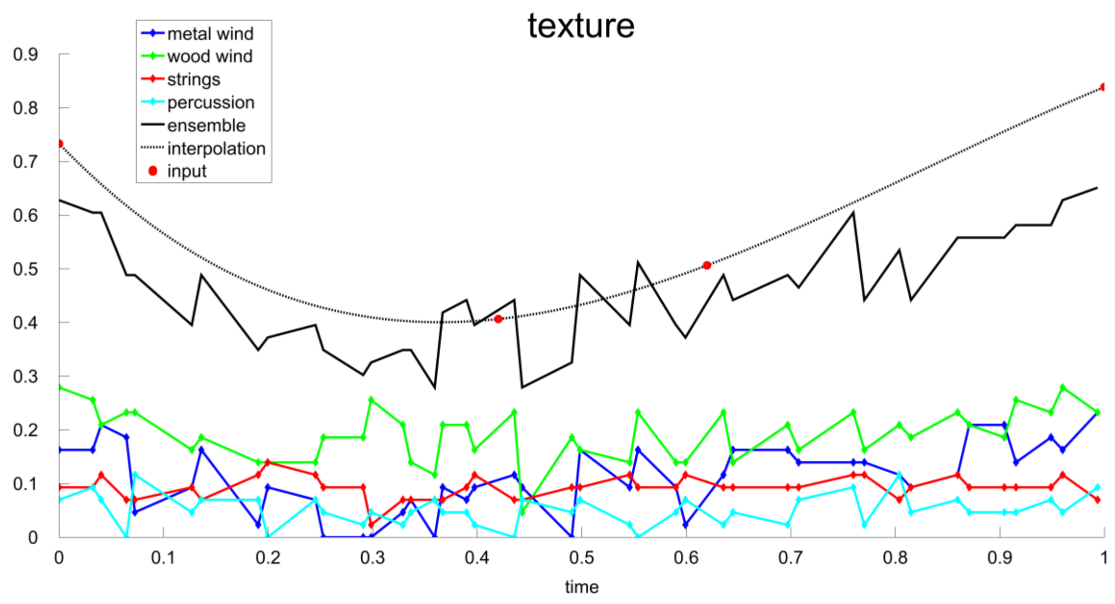


Figure 2.9. Interface to specify one of the compositional parameters. After the user places the desired points, the system interpolates them to obtain the whole curve.

A proposed goal in contemporary classical music was to produce a full album with different ensembles. With the help of the expert musician, we prepared seven settings to produce the corresponding sub-styles, whose result was the Iamus album, released in 2012 and containing the composition Hello World!; a piece for orchestra, which was recorded by the London Symphony Orchestra;[18] and some new chamber pieces for viola d'amore and harpsichord, clarinet, violin and piano, piano, violin, and voice and piano; recorded by recognized musicians in Sala María Cristina, Malaga.[19]

[18] http://lso.co.uk/

[19] https://es.wikipedia.org/wiki/Sala_Mar%C3%ADa_Cristina

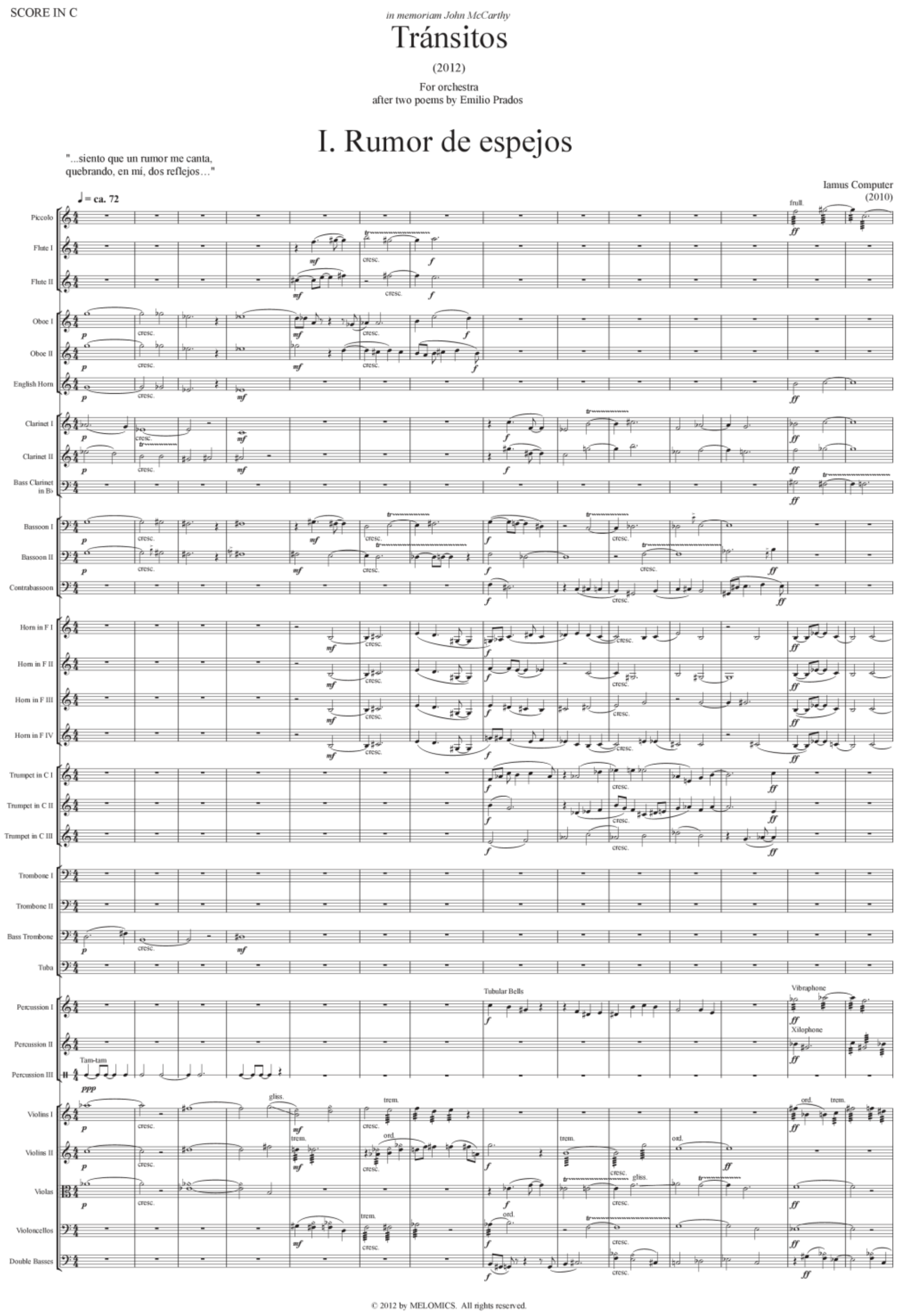


Figure 2.10. First page of the score of Transitos.

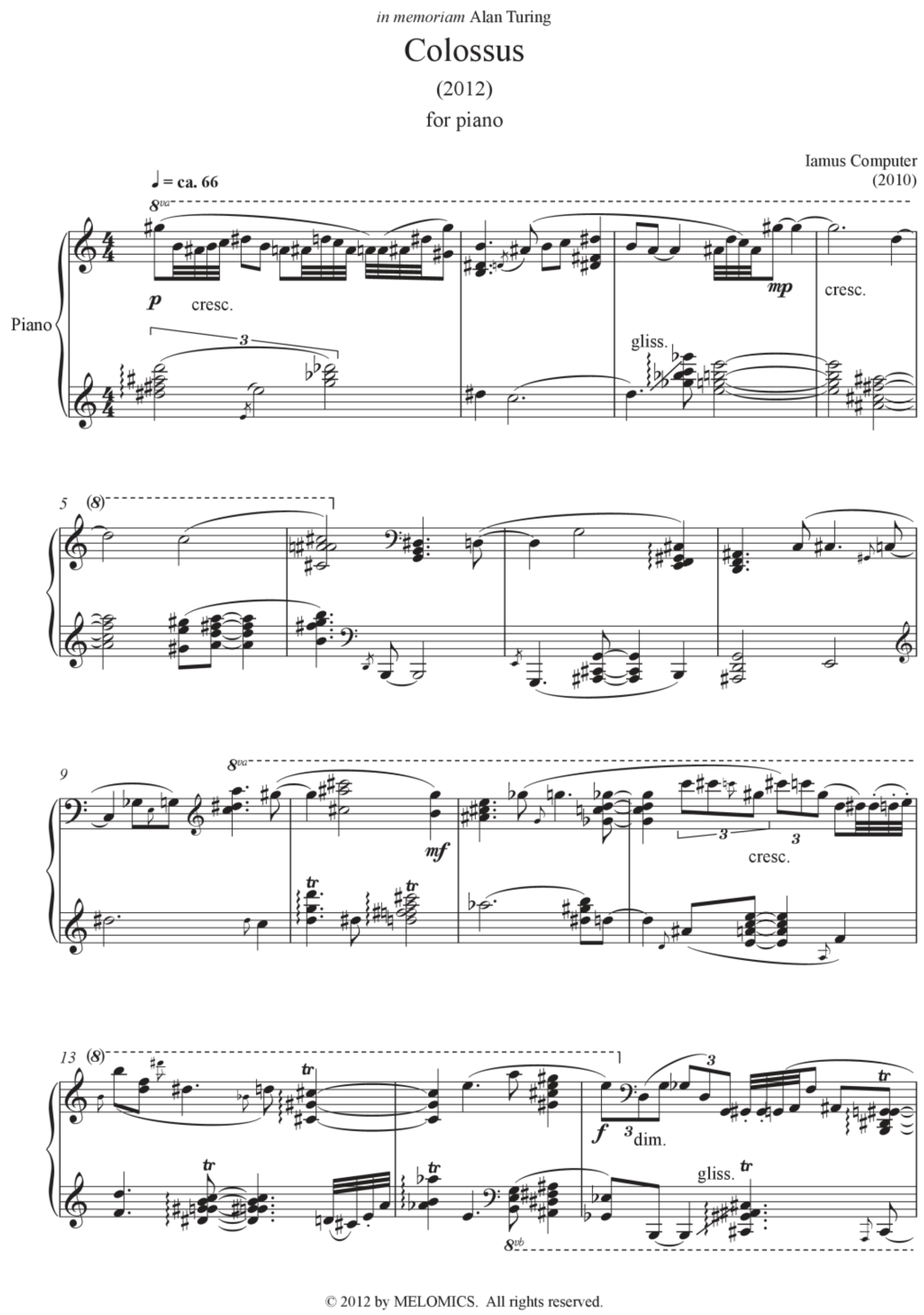


Figure 2.11. First page of the score of Colossus.

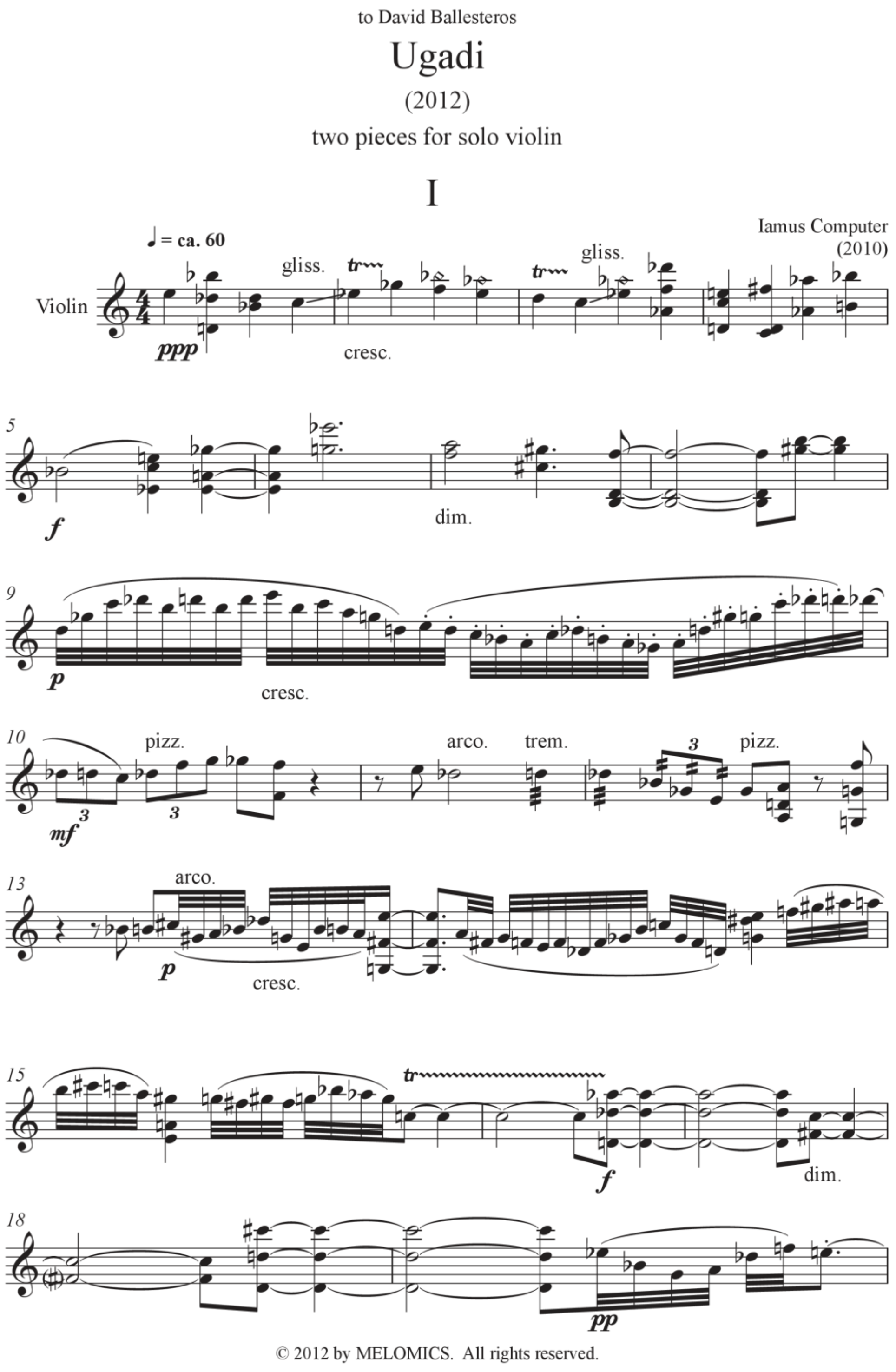


Figure 2.12. First page of the score of Ugadi.

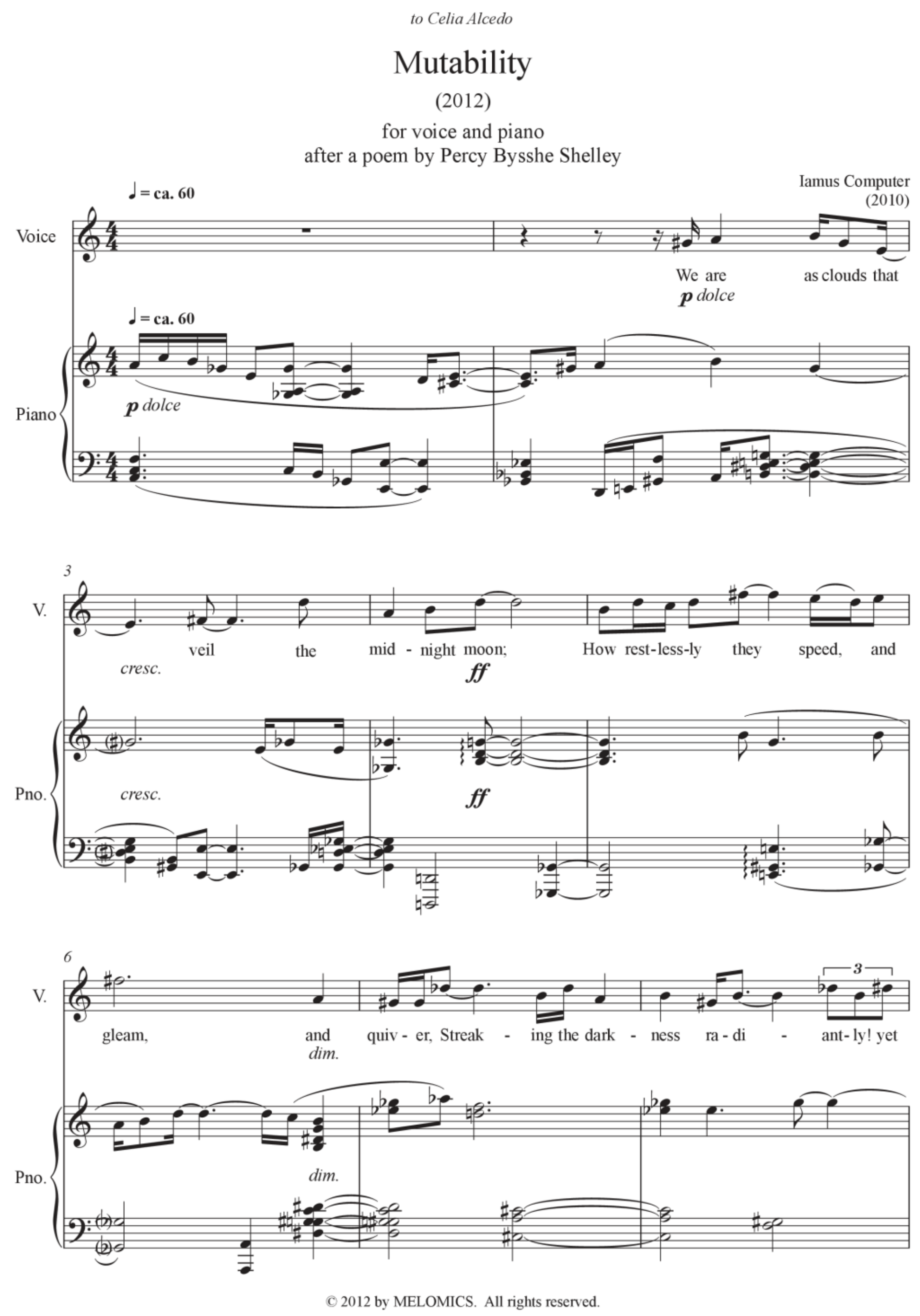

to Celia Alcedo
Mutability
(2012)
for voice and piano
after a poem by Percy Bysshe Shelley
Iamus Computer
(2010)
♩ = ca. 60
Voice
Piano
We are as clouds that
p dolce
veil the mid - night moon; How rest-less-ly they speed, and
cresc.
ff
V.
Pno.
gleam, and quiv - er, Streak - ing the dark - ness ra - di - ant- ly! yet
dim.
© 2012 by MELOMICS. All rights reserved.

Figure 2.13. First page of the score of Mutability.

At the same time, we used these musical settings to produce music in the computer cluster Iamus. During several days, it produced tens of thousands of compositions in the contemporary classical style that were stored in a web server and available in four different formats: MIDI, XML, PDF and MP3.

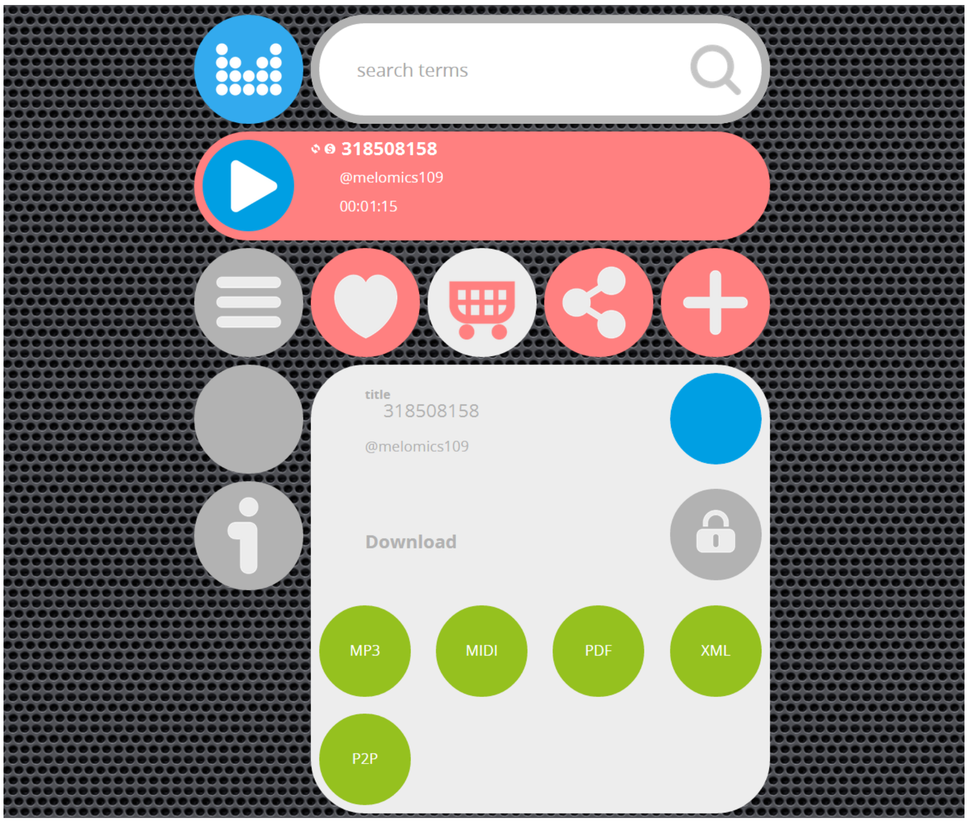


Figure 2.14. Melomics website. Page of a random composition.

2.2.3 Model for tonal music

After having created a number of compositions for many different purposes, the need to generate a more popular kind of music appeared. The so called atonal system would be able to produce tonal music, with the proper configuration. However, to produce it, the needed constraints will be too complex, and the system would have to run for too much time until achieving any result. Hence a different approach was followed; we adapted the genetic model into another that would favor the emergence of harmony and tonality.

As before, the new genotype is based on a formal grammar model, it undergoes a rewriting process leading to a string of symbols, which is interpreted into a matrix-based music structure, and the whole process is driven by a set of initial parameters and influenced by a physical context. To enhance the management of harmony, texture, dynamics, rhythms, polyphony and many other relevant musical elements, we introduced changes in the model at all levels; manly: we designed a new set of operational genes, we settle a more restrictive structure for the formal grammar, we implemented a collection of more advanced functions to make adjustments in the resulting string and we implemented a much more advanced mechanism to interpret the string, treating information in a much more abstract way than before. In addition, we developed a standardized tool that let us to specify musical styles, by introducing many different aspects of the music that is desired to be produced, as detailed in Section 2.2.3.1. In summary, we redesigned the front end of the system, with two stages clearly differentiated: (1) instantiating the musical style into a concrete genome, where random processes intervene and (2) developing the musical genome, being a deterministic process.

The new genetic model is able to manage the exact same musical parameters, but with a more control over them; besides, if the tool to specify styles is used with a loose configuration, the music obtained would probably not fit in any known style, and yet it would be perceived as harmonious or consonant. Despite introducing these constraints by default, especially if using the tool for musical styles, there is still a huge space of search available, in the same way as occur when a human artist is limited to produce music in a particular style.

In the next sections we will first describe the tool to build genotypes with certain features in common, which is a sort of template for creating musical styles, called the style-tool; then we will explain the formal basis of the model for tonal music, including the genetic representation of songs and the process of developing these genome into explicit data; and finally we will present some use-case experiments carried out with this compositional system that will serve as a timeline to illustrate the introduction of some new musical operators, a new relevant functionality or tool or a particular and interesting use of this technology.

2.2.3.1 Musical styles

To ease and standardize the way of specifying the music to produce, and also to help in reducing the execution times, we designed a tool based on filling a sort of questionnaire with a number of high level and low level musical parameters.

The information will be used to assess the developed pieces (mainly the parameters related to rhythm and harmony), but also to build the genotypes in a way that the corresponding developed pieces fit most of the specifications (we have tried to move most management of input parameters to this early stage, in order to improve the efficiency). Ultimately the style-tool has become the standard way to produce music with the tonal system. The parameters included and the way of using them to produce genomes was studied and designed with the help of experts. The nature of these inputs makes possible the interface to be handled by people with no knowledge at all about the way the software operates, but with the background to specify musical features; in fact, in most cases where a new type of music was needed, the style was completely filled solely by the musicians.

Next we will describe the categories of parameters included, classified in three sections.

Section 1: Global parameters

This section includes parameters that have an effect over the whole composition or that are used to build each compositional unit during the instantiation process.

- **Duration**. Boundaries or a specific value specifying the duration of the compositions.
- **Tempo**. A specific value or some lower and upper boundaries to choose the tempos to be used along a composition.
- **Dynamics**. A specific value or some lower and upper boundaries to choose the macro-dynamics to be applied along a composition.
- **Role types**. A list of ID corresponding to the roles or behaviors that may appear in the compositions. There are currently implemented 89 different roles, which can be classified in the next groups: melody, accompaniments, homophony (harmonized melody) and counterpoint. The harmonic and rhythmic accompaniments can appear in different forms: Chords, Bass, Pads, Arpeggios, Ostinati, Drums and Percussion.
- **Arpeggio configuration**. If the role arpeggio is enabled to appear, some additional parameters must be given: mode or contour type (Chord, Up, Down, Up-Down, Down-Up, Up-Down+, Down-Up+, Random); durations

or time segments into which the note values of the primary rhythmic cell must be divided; scope or number of octaves where the contour should be fitted into; and tie notes, indicating whether consecutive similar notes shall be tied or not.

- **Instruments**. For each role, there is a list of allowed instruments (plus additional specific properties) and their probability to be chosen.
- **Rhythmic incompatibilities**. Defines the incompatibilities of instruments to play notes at the exact same moment.
- **Scales**. A set of modes and notes of reference to build the musical scales to be used in the compositions. Resulting scales can be C-major, C-minor, D-Dorian or C-Mixolydian.
- **Harmony**. This category includes parameters to describe how to build the harmony in the different compositional levels; containing for example the allowed chords and roots, a measure of the allowed dissonance between melodies or the form of the chord progressions.
- **Rhythmic modes**. Includes the allowed types of measures, the types of accents to perform the notes and other related parameters.
- **Rhythmic patterns**. For each role is given a list of note values or patterns of them, establishing the desired frequency of occurrence in a composition.
- **Melodic pitch intervals**. For each role of the melody kind, a weighted list of pitch intervals is given to define how the melodic contour has to be built.

Section 2: Structure

The structure of a composition can be defined with a hierarchy of different levels (Bent & Pople, 2015). Our proposal counts on the next five levels: composition, period, phrase, idea and notes.

Below we describe the different sections to define the compositional structure using the style-tool.

- **Composition**. This is the highest structural level that can be defined. A composition will be formed by a sequence of similar or different kind of periods, possibly with some music operators (alterations in tone, harmony, tempo, macro-dynamics…) between each of them. Some of the parameters that can be established at this level (apart from the ones described in the previous section) are: boundaries for the number of periods, boundaries for the number of different periods, allowed music operators and boundaries for the number of measures in the whole composition.

- **Period**. This is the highest structural subdivision of a composition. There can be more than one type of period, which can be repeated along the composition. The different types are built independently, becoming separate musical units, recognizable in the composition. Some of the parameters that can be defined at this level are: boundaries for the number of phrases inside one period, boundaries for the number of different phrases, allowed music operators inside the period and boundaries for the number of measures in each period.
- **Phrase**. This is the third structural level, the constituent material of the periods. Some of the parameters that can be defined at this level are: boundaries for the number of ideas, boundaries for the number of different ideas, allowed music operators and boundaries for the number of measures in each phrase.
- **Idea**. Constitutes the lowest abstract level in the structure of a composition. Once again, a phrase can be composed by different ideas that can be repeated in time, with music operators in the middle. A musical idea will be a short sequence of notes generated independently for each role, using many different criteria (harmony, rhythm, pitch intervals, relationship with other roles…). Several of the described global parameters affect the process of creating an idea, and other applicable parameters at this level are: boundaries for the number of chords, boundaries for the size of chords and boundaries for the number of measures.
- **Texture**. This sub-section allows the possibility to define rules for the inclusion of roles; different types of dependencies between them; the compositional units where they are forced, allowed or prohibited to appear in; and general evolution of the presence of instruments.

Section 3: Special periods

It is possible to define types of periods with special behaviors. This section would enable to define for example musical fills or cadences, among other compositional functions. The parameters that can be set for each special type of period include the same as those used to configure the regular ones, plus a set of rules to define how they should be placed into the composition.

Introducing the hypersong

In order to handle the needs that came with the applications based on adaptive music (presented in Chapter 4), we implemented a new mode of running the style-tool, with two phases. First, some free parameters specified by the expert musicians are stochastically instantiated, as usual, resulting in a very narrow definition of the music that is going to be produced, a kind of meta-composition. In the second phase, the system will sequentially produce a variation of fundamentally the same composition, by instantiating the rest of musical parameters, where each combination of values are also specified. The resulting bundle of compositions, grouped as a single musical element, is called a *hypersong* (detailed in Chapter 4).

2.2.3.2 Model of genome

The model to represent tonal music derives from the previously designed and introduce four main changes: (1) The addition of new reserved symbols in the grammar, acting as music operators during the interpretation, to improve the management of harmony, with behaviors that include building chords, handling harmony along the tracks or taking harmony into account when building melodies. (2) The new symbols or genes count on a more abstract interpretation, making necessary a more complex function to translate the resulting string into the internal representation. (3) The production rules in the genotype are explicitly structured in levels, which are unfolded sequentially, with no possible backtracking, in order to settle a more direct connection to the hierarchical structure usually found in music, particularly in western popular compositions. (4) New parameters were included to guide the development of the grammar, the translation of the resulting string into the musical matrix, and to produce the output files; some of these parameters are part of the genome and some others belong to the compositional context that affect the developmental process.

In this model, the genotype can be better described with a deterministic context-free grammar:

$G = (V, \Sigma, S, P)$

where

V is the set of non-terminal symbols,

Σ is the set of terminal symbols (the alphabet),

S is the axiom or starting symbol,

P is the set of production rules with only one symbol on the left side and exactly one rule with each non-terminal symbol on the left side.

In the implementation, the set V can be identified with the structural units: periods, ideas… and Σ will contain symbols linked to the notes and the music operators, such as modulators of pitch, duration, current harmonic root or current chord.

Next we present some of the reserved terminal symbols used in the tonal model and their musical interpretation:

N increases the counters pitch and harmonic root in one unit.

n decreases the counters pitch and harmonic root in one unit.

[saves in a stack the current value of pitch, harmonic root and duration.

] restores from the stack the last value of pitch, harmonic root and duration.

$<$ saves in a stack the current time position, value of pitch, harmonic root and duration.

$>$ restores from the stack the last saved time positon, value of pitch, harmonic root and duration.

$W4.0$ applies the macro-dynamic mezzo-forte.

$M0.0.0.0$ makes the next symbol linked to an instrument, to interpret the root note of the current chord, instead of the current computed pitch.

For an extensive description of the reserved symbols and their musical interpretation, see Appendix A.2.

Next we give the definition of G_t, a simple example of grammar to illustrate the instantiation of this model:

$G_t = (V, \Sigma, S, P)$

$V = \{Z, A, B, C, D, E, F\}$

$\Sigma = \{N, n, [,], <, >, W4.0, a, b, s, M0.0.0.0\}$

$S = \{Z\}$

and P consisting of the following rules:

$Z \rightarrow W4.0[ANNNNNNNA]B$

$A \rightarrow CC$

$B \rightarrow D$

$C \rightarrow ENEnE$

$D \rightarrow FFNF$

$E \rightarrow < anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b$

$F \rightarrow as[NNNa]a$

The terminal symbols a, b and s correspond to notes to be played by their associated instruments, s typically being the musical rest.

The interpretation of this grammar will be detailed in the next section. Nevertheless, we can bring forward some conclusions. The higher structural level is represented by Z, the axiom. It will develop into two identical musical units (periods), represented by the symbol A, separated by seven steps in the pitch dimension (meaning of symbol N), and then followed by the period B. The two types of periods A and B would develop into two simple sequences of phrases, CC and D respectively. Each phrase C and D consist of a sequence of 3 ideas. Idea F is interpreted only by the instrument linked to a and idea E will be performed by instruments associated to a and b, in polyphony; the latter always playing a harmony consisting of the root note of the current chord.

The structure and global parameters of the genomes are usually built with the previously described style-tool, following a compositional hierarchy of five levels; the right side of the production rules are built in the same fashion than in the atonal model; and, in spite of all the new constraints and biases to produce harmonic order, the model is still intended to preserve the original advantages: readability, robustness to alteration, high musical expressiveness and capability of creating original and useful results.

2.2.3.3 Model of development

The first stage in the genomic transcription is the rewriting process of the context-free grammar, leading to a string of characters with no non-terminal symbols remaining, typically after four iterations between the five levels. Below we will illustrate the resulting string during the development of the grammar G_t defined in the previous section (using spacing and brackets for clarity):

Iteration 0 (composition)

Z

Iteration 1 (periods)

$W4.0[ANNNNNNNA]B$

Iteration 2 (phrases)

$W4.0[(CC)NNNNNNN(CC)](D)$

Iteration 3 (ideas)

$W4.0[$

$((ENEnE)$

$ENEnE))$

$NNNNNNN$

$((ENEnE)$

$ENEnE))$

$]$

$((FFNF))$

Iteration 4 (notes)

$W4.0[$

$(((< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)N(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)n(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b))$

$((< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)N(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)n(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)))$

$NNNNNNN$

$(((< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)N(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)n(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b))$

$((< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)N(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)n(< anaNNsa > M0.0.0.0b\ M0.0.0.0b\ s\ M0.0.0.0b)))$

$]$

$(((as[NNNa]a)(as[NNNa]a)N(as[NNNa]a)))$

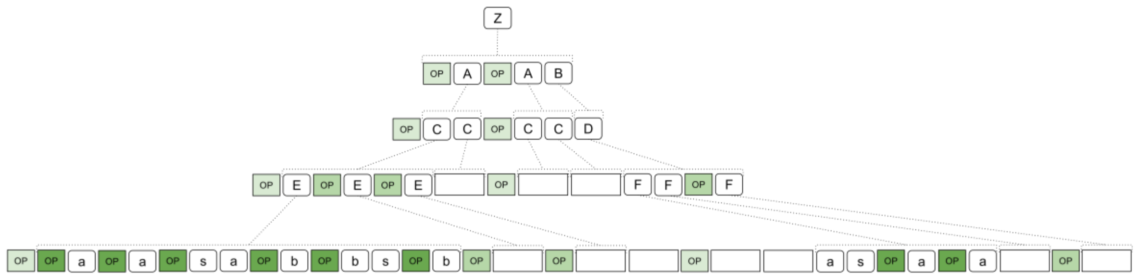

Z
OP A OP A B
OP C C OP C C D
OP E OP E OP E OP F F OP F
OP OP a OP a OP s a OP b OP b s OP b OP OP OP a s OP a OP a OP

Figure 2.15. Example of tonal music. Developmental tree. The operators have been summarized as "OP".

The resulting string will suffer additional adjustments, as well, before being interpreted into the matrix structure. Some of these modifications are aimed to clean the string from useless genes, for example deleting the consecutive repetition of an idempotent music operator. Some other readjustments are caused by symbols representing genetic regulators. These functions, acting according to different criteria, could be adding or suppressing symbols for musical effects, duplicating or deleting certain substrings, splitting or merging a sequence of notes, or rewriting certain substrings to acquire final instantiated values.

The second stage of the development is to interpret the final string into a matrix structure with the explicit values of each note and track. As in the system for atonal music, the process will consist of a sequential reading of the string from left to right, with a general behavior based on shifting the current musical value of each parameter (pitch, duration, chord, dynamics…) when the corresponding operator appears. A musical note is created every time a terminal symbol associated to an instruments is read, acquiring the present value of the musical variables. These mechanisms are more complex than before, for example, the actual note pitches are not computed until this phase, since the musical scale changes dynamically.

In Figure 2.16 is illustrated the interpretation of the resulting string from the previous example. As initial values, we use the C major scale, a default duration of quarter note, a tempo equal 80, a default dynamic mezzo-piano, an initial pitch MIDI 60, an initial chord major triad, an initial root I, the instruments violin and double bass for symbols a and b, and s as the musical rest. The violin plays a single melody and the double bass is the accompaniment (in parallel due to the time operators <, >), always playing the root note of the harmony, which changes between ideas and increases one octave between the two repetitions of the period A. The double bass will be playing one octave lower because of its internal configuration of tessitura, beginning in MIDI 48 instead of the default MIDI 60. As we can observe, the genome produces small pitch shifts between ideas, no pitch movement between phrases and a big jump in the scale between the first and the second periods. The second kind of period (the third total period) is only played by the first instrument, with a complete different structure than the first.

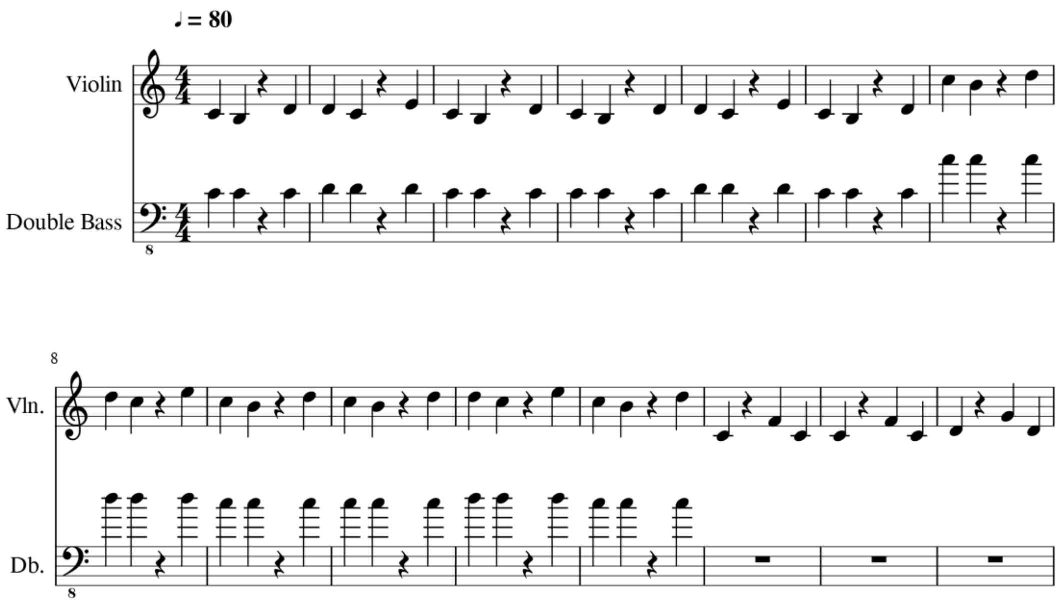


Figure 2.16. Example of tonal music. Resulting score.

- Sample 2.19. Tonal example. Resulting composition.

The music content and the meta-information will be stored in the internal representation, described in Section 2.3.1. This format was extended to serve the tonal system, mainly, on the one hand, to store meta-information regarding the compositional structure, the musical style, the instruments and other data about the decision making during the composing process, in order to enhance the storage, management and search of these themes in the web server. On the other hand, there was introduced additional data about the concrete chosen instruments, the performing techniques, the effects and the mixing and mastering parameters, since the main output format targeted now is the synthesized audio, namely the MP3, instead of the symbolic format. As in the atonal system, the information written in this structure is more manageable to carry out the final adjustments caused by the context (tessituras, specific effects and dynamics, or even a final readjustment of pitches) and to produce the rest of all possible output files.

2.2.3.4 Relevant experiments

Genetic engineering: SBS Logo

As a showcase of a powerful application of this tool based on genetic representation, we did an analogous experiment to the one with the Nokia tune in the atonal model. The system was going to be presented at the event Seoul Digital Froum 2015 (SDF2015) hosted in Seoul and it was going to be broadcasted by the Korean SBS TV channel, so we decided to make use of their musical logo[20] in our next experiment.

The first stage was to reverse engineer the musical piece. This time we counted on the help of an expert musician to analyze all the possible aspects that could be handled with the tonal system: different structural levels, global harmony, chord progressions, rhythms, melody and intended evolution of these parameters. We wrote the artificial genome increasingly, first encoding the structure and the global parameters and then adding each role, one by one, until we got a piece that we considered to be the base SBS Logo, formed by a melody, a bass and a pad.

The second stage was to generate a family of themes, with the SBS Logo as the central relative. We wanted to illustrate three uses:

- A possible evolutionary process that could have been occurred in our system, from a very elementary piece of music, to give place to the logo. We obtained this path reversely, by simplifying the genotype manually. First we deleted the melody, secondly we simplified the structure, and then we removed the bass role, remaining just an instrument playing a simplified version of the original harmony.

[20] https://www.youtube.com/watch?v=GdwU2qkEbRE

- Generating a variety of themes by mutating the SBS Logo, continuing the phylogenetic evolution that had been started. Many random but coherent mutations were executed, including duplication of structures, altering pitches, altering progressions (root, chords), changing global parameters (tempo, dynamics, scales…), commuting roles, changing instruments, adding new kind of roles (percussions and harmonizations), adding effects, and changing the textural evolution (in Figure 3.2, which illustrates the study of distances between relatives of the SBS Logo, some of the evolutionary paths are shown).
- Contaminate the genome of an independent musical theme with genetic material from the SBS Logo, preserving the identity of the original theme but being possible to identify the logo as part of it. We produced some compositions in one of the musical styles that we had designed and we tried several ways of introducing low level genetic material from the SBS Logo, replacing parts of the low level structure of the pre-generated compositions. One example (shown in the phylogenetic tree of Figure 3.2 as "d" and in Sample 2.23), consist of a theme in the style Disco02, where one of the present ideas, despite having been shortened and altered to fit in the host theme, can be recognized as coming from the SBS tune.

- Sample 2.20. Audio of SBS Logo reengineered.
- Sample 2.21. Audio of SBS Logo relative 1.
- Sample 2.22. Audio of SBS Logo relative 2.
- Sample 2.23. Audio of SBS and Disco02 crossover (human synthesis).

Chronic Pain app

The standard method for creating arrangements of music content for adaptive applications, the hypersongs, was designed with the app eMTCP. Continuing the work started with the atonal system for application to reduce anxiety, we included these mechanisms in the style-tool interface (as commented in the section *Introducing the hypersong*). This implied not only the logic to generate a bundle of related genotypes (a hypersong), but also a function that produces altered versions of these genotypes, because each version needs to be fragmented in time as well. Further explanation on this app, feedback-based music generation and more examples will be given in Chapter 4.

Web repository and 0Music album

In 2014 we aimed to populate the web repository with music from the tonal system, as it was done before with the contemporary classical style. The target was to produce pieces of music of more popular genres, using several of the styles that we had designed so far, including disco, minimalist, relax, modern band, world music, synth music, pop, fusion and symphonic music. About fifty thousand themes were made available in the website, so the users could search in this space of music, fully acquire the desired themes at a zero cost (under a CC0 license) and download them in the different formats.

- Sample 2.24. Tonal music from the web repository. Example 1.
- Sample 2.25. Tonal music from the web repository. Example 2.
- Sample 2.26. Tonal music from the web repository. Example 3.

Generating this collection of themes served also to test the current capabilities of the tonal system, especially (a) in controlling harmony in compositions with many interrelated roles and (b) in producing enough diversity of themes in these styles, despite the amount of constraints imposed. To show-case the current state of the system, a small sample was selected by musicians from the repository. They choose a collection of themes that they considered representative of the different styles that the algorithm could produce. As a result, the 0music album was released during a one-day symposium at MIMMA, Málaga, on July 21, 2014.[21]

[21] www.geb.uma.es/0music

Figure 2.17. Search tool of the website and 0music album cover.

- Sample 2.27. 0music album.

Styles DocumentaryPop and DocumentarySymphonic

In 2014 we contacted a company that wanted us to produce music for media productions, such as documentaries. After a period of iterations to settle the music specifications, we designed two styles: one to generate classical music and another to produce a more popular kind of music. This use case has been included to highlight the fact that there was no need to implement any new functionality in order to satisfy a demand of a new kind of music, we just had to fill the proper values in the style-tool and the resulting output was indeed valid. We produced 200 musical themes, 100 of each style, in both high quality MP3 and uncompressed WAV, having been synthesized, mixed and mastered automatically.

- Sample 2.28. Theme from the style DocumentaryPop.
- Sample 2.29. Theme of the style DocumentarySymphonic.

2.3 Music notation

One of the main targets while building the system was to be able to store all the generated information in a standardized and easily readable internal format. In the system for atonal music we had to use this information mainly to build the music score (and other symbolic formats), which should be readable, or loadable into a composition editor, to be analyzed, edited and exported to other formats. In the tonal system we also needed to manage the symbolic information properly, in order to produce the main targeted format, the synthesized audio (emulation of the musician's interpretation).

2.3.1 Internal representation

The internal format is the way the symbolic information is stored in the system, after a genome having been interpreted. Despite having evolved with the development of the two versions of the compositional system, the internal representation maintains its fundamental structure, consisting of (1) a series of global fields with values affecting to the whole composition or describing it:

- Composition name
- Composition ID
- Composition logo
- Name of the style which gave place to the composition
- Name of the author
- Date of creation
- Structure with the input values and ranges, and their final instantiation
- Structure describing the development of the genotype
- Structure describing the instrumental texture
- Tags and keywords associated to the composition (used for example when they are stored in the web server)
- Initial and default values of musical variables (tempo, key, macro-dynamic...)
- Database of the virtual instruments used
- List of output formats

(2) A set of lists containing information associated to each track:

- Name of instrument
- Role ID
- Instrument ID
- Instrument type ID
- Index of virtual instrument in the database
- Set of effects and configuration

And (3) a 3D structure, where the first dimension correspond to the tracks, the second dimension is associated to the time and the third one contains each property that every note in a composition has, which includes:

- Onset
- Duration
- Sound or rest (binary field)
- Pitch
- Dynamic, velocity and volume (related parameters whose application depends on the targeted output)
- Note value and representation details for the score (keys, bars, note stems…)
- Instrumental effects and ornaments
- Subjective directions

2.3.2 Exchange format

The need to communicate certain information about the compositions to another external system, and not the compositions themselves, appeared eventually. An example was the use of Melomics music in an application with dancing avatars. These actors followed the leading beat of the music, but also changed their choreography according to compositional features like structural or textural evolution, which had to be provided separately. Storing compositions in the web server is another example that required a way to include some additional information.

To satisfy these needs, we designed an exchange format containing meta-information about the compositions, consisting of a text file based in the open standard JSON[22]. To date, the fields that have been includes are: the composition name, the style, applicable tags or keywords, the effects used on each track in the synthesis of the MP3, the hierarchy of structural entities, the global bpm, the textural evolution of instruments and the number of measures.

2.3.3 Output formats

For a single composition, Melomics system produces more information than any usual music format is able to handle. This is because the information has to be enough to produce all of these other formats (also because with a computer we are able to save more information than the human composers usually retain about their proceedings). According to the nature of the stored data, a music format can be of two types: (a) symbolic, representing abstract information, like pitches, tempos, durations, etc. and (b) audio, with the music being handled as sound waves. Examples of the first kind are: MIDI, LilyPond, MusicXML, Open Sound Control[23] or the proprietary formats used in notation software like Sibelius®, Finale® or Notion™. Examples of the second kind are WAV, FLAC, MP3 or OGG[24] and also platforms like Open Sound System[25] or Csound.[26]

In Melomics we have targeted three main output layouts, described next.

2.3.3.1 Score

The musical score is the final target in the system for atonal music (Moreno-Arcas, 2011). We used the MusicXML format, despite being proprietary, because

[22] http://json.org/
[23] http://opensoundcontrol.org/
[24] https://xiph.org/ogg/
[25] http://www.opensound.com/
[26] http://csound.github.io/

it is fully documented, can be used under a public license and it is compatible with most of the score writing software, digital audio workstation (DAW) and other music tools, including: Sibelius, Finale, Notion, MuseScore,[27] Cubase™, Logic Studio®, Mixcraft™, Reaper or LilyPond.

The functions that translate the information from our internal format to the score in MusicXML are quite simple and mostly centered on formatting issues, since the required data is already computed and made explicit. In order to have a musical score ready to be interpreted, the system also generates the corresponding printed score in PDF format. The Lilypad[28] tool is used to convert the XML into a LY file, which is then printed in PDF with LilyPond.

2.3.3.2 MIDI

The MIDI, commonly used with real time virtual synthesizers, is a kind of symbolic music format. It comprises a more reduced set of parameters than the MusicXML can handle, but we have nevertheless considered it as a main output because:

- Its use as an exchange format is wider extended than the MusicXML, in score writing and many other music tools, especially to import samples into DAW.
- It is the most usual format to perform symbolic music information retrieval. It is used for example in jSymbolic, a component of the jMIR suite (see Section 3.1 and Section 3.2).
- The automated music synthesis in Melomics makes use of the MIDI format as input, plus some additional parameters.

In the primary versions of the system, the MIDI files were obtained from the XML, through LilyPond, the same as with the PDF. Currently, we use a python library (MIDIUtil[29]) to directly build the MIDI file from the internal representation, which makes the process simpler, faster and more controlled.

[27] https://musescore.org
[28] http://lilypond.org/download/gub-sources/lilypad/
[29] https://code.google.com/p/midiutil/

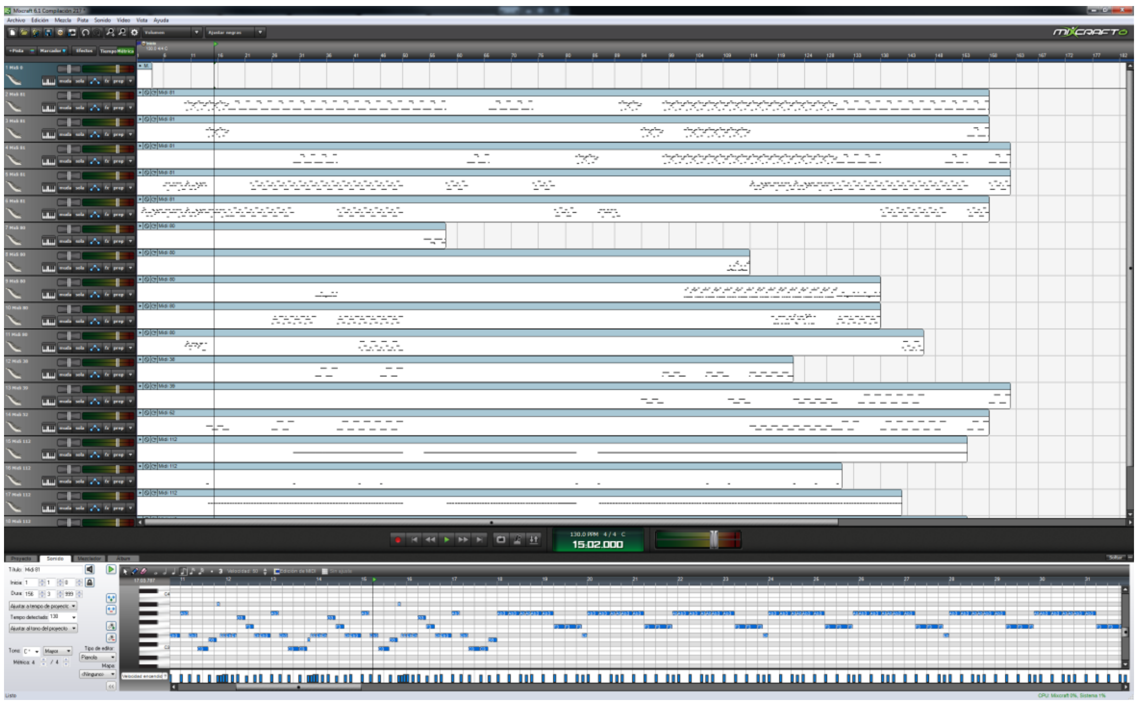


Figure 2.18. MIDI file loaded in the digital audio workstation Mixcraft.

2.3.3.3 Synthesis

In the compositional workflow, the most advanced way of presenting the music is the audio format, since it requires all the symbolic information to be produced in advance. The synthesized audio is an approximation of the sound waves deriving from the generated symbolic content (pitch, volume, durations guidelines for instrumental performance...). The software was able to produce WAV files from the internal representation, making use of synthesizers like FluidSynth[30] or TiMidity and virtual instruments in the SoundFont 2 standard (SF2) (De Vicente Milans, 2013). Then, the WAV file can be converted into any other audio format (FLAC, MP3, OGG...) with common tools. This component of Melomics was further continued, in order to get closer to the results that could be achieved by an expert musician using a DAW to do the same job. In the newer version, the WAV file was obtained with a scriptable DAW called Reaper, which enabled a more complex editing and managing VST, a more advanced standard of virtual instruments than SF2.

[30] http://www.fluidsynth.org/

2.4 Summary and discussion

2.4.1 Composing process

Systems for atonal and tonal music, although having different genomic representation and actually being used as separate software, they share the exact same code structure and execution workflow, which can be described as follows:

- Filling the desired input parameters in the designed interface. In any case the parameters will represent musical specifications or directions at different level of abstraction, with no need for creative input from the user.
- Some of the input parameters will be used to drive the stochastic processes of producing genomes.
- A genotype, based on deterministic grammar and stored in a plain text file, is read and developed into a resulting string of symbols.
- The string of symbols is rewritten after some processes of adjustment caused by internal and external factors.
- Each symbol in the final string has a low level musical meaning, which is interpreted by the system through a sequential reading from left to right and stored in an internal representation.
- Once again the musical information will be adjusted and stabilized due to internal and external factors.
- Some of the input directions will be assessed and different actions might be taken as a consequence: the current composition may be discarded, it may simply pass the filter or it could pass the filter and some of its genetic material being saved for future executions.
- Finally, the different modules for translating the musical information to the corresponding output formats may be used.

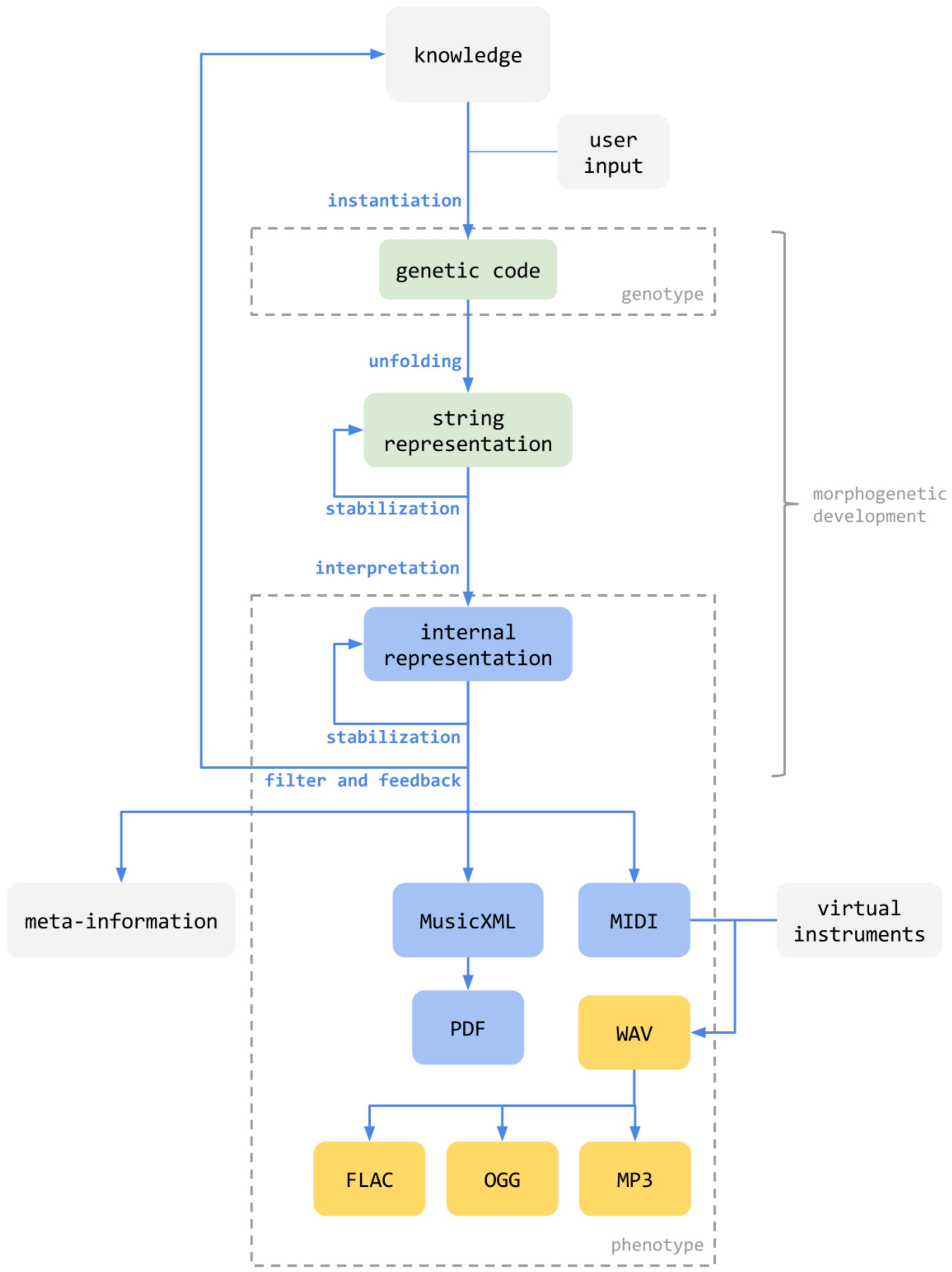


Figure 2.19. Representation of Melomics global workflow.

The spent times in the described workflow may vary depending on some factors, such as:

- The number and kind of targeted outputs will have a considerable effect. For example, obtaining the MP3 is computationally expensive by itself, but besides, it requires the corresponding MIDI and WAV to be generated in advance.
- The size of the ensemble of instruments will notably affect as well. A big ensemble will make the process slower at any stage, for example in producing the genome, since more interrelations must be considered, in its development, since it is bigger, or in the production of the synthesized audio, since more tracks have to be processed.
- The duration of a composition is another factor that increases (proportionally) the time spent at any level of the workflow.

In general, the times spent in producing compositions with the atonal and the tonal systems are similar, since they are integrated by equivalent phases and processes in the workflow. A study of execution times, with concrete use-cases, will be given in Appendix C.

2.4.2 Main technologies used

Among the software employed during this thesis we highlight some of them:

- **MATLAB® R2010b**[31] is the language and developing environment used to code most of the algorithms, including the core system. We mostly use the standard libraries, but for certain tasks we also take advantage of some additional toolboxes (parallel or computer vision, for example).
- **Python 2.7**[32] was employed to support the backend of the tonal system, from interpreting the resulting string of symbols (after the genome is unfolded) to produce the different output files.
- **Lilypond**[33] is a suite for score writing. We made use of its format (.ly) and software, mainly as a middle format between the XML and the MIDI and PDF.

[31] www.mathworks.com/products/matlab/
[32] https://www.python.org/
[33] http://lilypond.org/

- **TiMidity++**[34] was used in the atonal system to synthesize MIDI into a WAV file.
- **SoundFont® 2**[35] is the virtual instrument technology used to synthesize with TiMidity.
- **Reaper**[36] is a graphical DAW with the possibility of programing scripts to automatize editing functions (in Python language). We use this tool with the latest version of the synthesis module.
- **VST™ instruments**[37] (VSTi) is a virtual instrument technology, more advanced than SF2, which we used with Reaper in the latest version of the synthesis.
- To convert WAV to other synthesized audio formats we have made use of the standard tools **ffmpeg**,[38] **lame**,[39] **sox**[40] and **flac**.
- **Mercurial**[41] has been used as a revision control tool during the development of Melomics.

2.4.3 Results

In this chapter we have exposed the foundations of Melomics technology, a tool which, during its more than six year of continued development, may be considered a noteworthy contribution to the field of algorithmic composition. It has achieved a number of milestones on the road, and some of them are summarized next: in 2010, a start-up proposal based on adaptive music therapy through Melomics is awarded by the University of Málaga;[42] later, the first piece on its own style (Opus #1) is created; MELOMICS research project is funded by the Spanish ministry of science; in 2011, the full work Hello World! is premiered; an article about composing with Melomics is published in Leonardo Music Journal by Professor Gustavo Diaz-Jerez (Diaz-Jerez, 2011); in 2012, Iamus album pieces are created and after recorded by the LSO and other top-shelf

[34] http://timidity.sourceforge.net/
[35] http://www.synthfont.com/sfspec24.pdf
[36] http://www.reaper.fm/
[37] http://ygrabit.steinberg.de/~ygrabit/public_html/index.html
[38] https://www.ffmpeg.org/
[39] http://lame.sourceforge.net/
[40] http://sox.sourceforge.net/
[41] https://mercurial.selenic.com/
[42] http://www.uma.es/emprendedores/cms/menu/spin-OFF/spin-2010-2011/

musicians; a live concert premiering works from Melomics is celebrated at the ETSII of the University of Malaga and a great media attention is shown during these years;[43] Melomics technology makes #70 in the 100 top stories of Discover Magazine (Berger, 2013); Melomics Hackathon is celebrated at UC Berkeley[44] and Melomics' technology is presented at Googleplex during the SciFoo[45] session *music-as-it-could-be*; Melomics music is used in therapeutic applications and experiments, and the eMTCP app is recommended by the American Chronic Pain Association;[46] in summer 2014 the second album 0music is released after a one day symposium; Melomics music is tested against human composed music at MIMMA and the music conservatory of Malaga; in the middle of 2015 Melomics is presented in the event SDF2015; and, in march 2016, four new pieces are premiered in MaerzMusic,[47] a festival hosted in Berlin.

The designed computational methods have allowed us to produce thousands of original compositions. Any of these can be randomly altered into consistent mutations, or modified in a directed fashion to get a specific version of it. We can reuse the genetic materials to produce new, possibly more complex, compositions, or to create contaminated versions of preexisting pieces. The proposed genetic models are at least capable of representing any kind of music that has been required, in the same way it can be done with standard music notation. At the present, the system also includes a set of tools and the musical knowledge to produce music in a wide range of styles. Apart from the music produced with specific purposes or the one made available to the general public through the website, Melomics music is used in other contexts, such as in therapeutic systems and mobile applications. Additionally, it is intended to map the defined style-tool into a sort of online questionnaire, so the community of user can interact directly with Melomics and be able to design their own new styles.

[43] melomics.uma.es/news
[44] www.melomics.uma.es/melockathon
[45] http://www.digital-science.com/events/science-foo-camp-2013/
[46] http://theacpa.org/music-app
[47] https://www.circus-berlin.de/maerzmusik-2016/

Chapter 3

Properties of computer generated music

This chapter is focused on the study of the music created by Melomics. We first show a series of experiments performing a quantified analysis of the musical features on both the symbolic and the audio formats. Secondly we present a study from a more subjective perspective, facing Melomics music against human made compositions.

The chapter is organized as follows: In Section 3.1 we introduce the concept of Music Information Retrieval (MIR) and describe some the best known MIR tools and methods; then we present some analysis on the music coming from Melomics (sections 3.2.1 and 3.2.2). In Section 3.3 we describe an experiment aimed to measure the valence of computer music compared to human music, and to check whether human listeners are able to identify Melomics compositions as made by computer (Section 3.3.1). Then we show some other experiences with Melomics music, from a subjective point of view (Section 3.3.2). Finally, in Section 3.4, we expose the conclusions of all these studies.

3.1 Music Information Retrieval

MIR is an interdisciplinary field of knowledge dedicated to obtain information from different kind of musical data. The most known and obvious source of data is an audio recording, from which the information (such as the musical style or gender, the participating instruments or voices, the arousal and valence, or the type of rhythms performed) is gathered after analyzing the wave signal. Nevertheless, the data sources subject to MIR analysis comprises the whole variety of formats, including also symbolic standards and even cultural information related to a musical piece (provided by people in websites or social networks, for example) (Lamere, 2008) (Schedl, Gómez, & Urbano, 2014). Some of the most common applications of MIR are recommendation systems; music transcription systems, to reverse audio recordings into scores; theme recognition tools, to identify songs or related versions; and categorization tools, to arrange songs according to the musical properties. In the present, most of the commercial music tools and services (Spotify®, vevo™, rdio™, PANDORA®, Shazam™...) include some kind of MIR application, either as part of its main functioning or at least in their recommendation systems.

In research there has also been a growing interest in the matter, for example through ISMIR,[48] an international organization, coordinating conferences and committed to promote MIR; also with an increasing number of publications, to cite some of them: the algorithm of Shazam (Wang, 2013), MIR techniques based on cultural information (Lamere, 2008) (Schedl, Gómez, & Urbano, 2014) or a doctoral thesis describing a complete toolbox for MIR, called jMIR (McKay, 2010).

Below we describe two well-known MIR suites that we have used to study Melomics music:

- The Echo Nest™.[49] It was a company acquired by Spotify in 2014, owning one of the world's greatest platforms to provide with music information to developers and media companies using MIR. It has a huge and continuously growing database of analyzed songs and artist. Globally, their main services include (a) helping to discover new music that matches certain desired parameters, allowing a high personalization of playlists as well; (b) providing updated and extensive information about any musical content that they have processed; (c) giving support for fingerprinting of musical content; and (d) providing with music editing tools. To developers, they also give access, on the one hand, to external

[48] http://www.ismir.net/
[49] http://the.echonest.com/

sandboxes, making possible to build applications by exploiting these services from third parties; and, on the other hand, they provide an API and some libraries in different programming languages, to interact with the information stored in their database. The web-based API contains a collection of methods that respond in the JSON or XML formats. These methods are grouped into (1) artist methods, providing biographies, related images, genres, terms, songs, reviews, hotness and others; (2) genre methods, with artists, profiles, similarities, etc.; (3) song methods, to access the gathered information and the analyzed musical features, such as loudness, hotness, danceability, energy, mood and countless others; (4) track methods, for analyzing or getting information about the tracks of a piece of music; and (5) playlisting methods, to allow managing playlists and user profiles.

- jMIR.[50] It is an open-source application for MIR implemented in Java, mainly used in research contexts. Its architecture is highly compartmented and its modules communicate with each other through the ARFF format (Witten & Frank, 2005) used in WEKA[51] or through the built-in ACE XML. The system's core is divided in two blocks: (1) components to obtain features, including jAudio, to analyze audio waves (compactness, peak detection, moments, strongest frequency…), jSymbolic, to study the symbolic MIDI format (instrumentation, texture, rhythm, dynamics, pitches, chords…), jWebMiner, to extract cultural information from the web (author, genres or styles, similarities between songs or artists…) and jLyrics, for mining information from lyrics (number and frequencies of words, syllables and letters, number of lines, punctuation, misspelling, vocabulary richness…); and (2) components to process the obtained information, specifically jMIRUtilities, to combine the extracted features, labelling and performing other miscellaneous tasks, and ACE, to execute a variety of classifications depending on each particular problem.

[50] http://jmir.sourceforge.net/

[51] http://www.cs.waikato.ac.nz/ml/weka/

3.2 Analysis of the generated compositions

We have run two types of tests: On the one hand, we wanted to know how the system really managed the relationship between codified and resulting music, studying how the changes in the genotypes would have an impact into their corresponding developed forms. Especially we wanted to check whether small mutations in the genome will produce small changes in the resulting theme, as expected, and also whether genotypes from a certain input configuration (expected style) actually produced music that could be classified into the same musical style. On the other hand, by using the same analytical tools, we studied where Melomics music could be placed in a space of musical parameters, in relation to the preexisting human music and to other kind of sounds.

3.2.1 Measurement of similarity

We choose jMIR to study the features in Melomics compositions, in particular its component jSymbolic. It operates over the MIDI standard, by extracting 111 different features, into the categories of instrumentation, texture, rhythm, dynamics, pitch statistics, melody and chords; including both single-value and multi-dimensional features. Most of MIR tools work on the audio wave (such as jAudio, from the same toolbox), partly because it provides with additional information about the actual piece under analysis (performance attributes, etc.), but mostly because music is usually found in this type of formats. However, analysis on symbolic notation is more adequate to get information of a high level of abstraction, more meaningful in compositional terms, which is best suited to our purpose.

Among the compositional characteristics that jSymbolic can extract, we have selected all of the single-value features and some of the multi-dimensional: "Fifths Pitch Histogram" and "Pitch Class Histogram", which provide with some information about tonality and harmony; and "Pitched Instruments Present", indicating which of the MIDI instruments are used. Since many of the features are not normalized, we have designed some simple methods to assess distances in this space, based on obtaining first the centroid of a set of themes and then giving measures relative to this point.

3.2.1.1 Compositional analysis to assess variations produced by mutation and crossover operations

A first study was made using the Melomics subsystem for atonal music. We took the bundle of compositions that resulted from the experiment in genetic engineering described in Genetic engineering: Nokia tune, with the aim of checking whether a small set of changes in the genome, yields a similar composition with respect to the original, and how the compositional distance increases as the set of mutations and crossovers becomes greater too. To assess how the resulting phenotypes differ from the original theme, we have compared each theme with the bundle's centroid and with the original tune. Concretely, for a given feature, if the current theme being evaluated is further from the Nokia tune than it is from the centroid, that would add one unit to the measure of distance for this theme (see the equation below).

$$d_{current} = \sum_{i=1}^{n} abs(Reference_i - Current_i) > abs(Centroid_i - Current_i)$$

where

$d_{current}$ is the computed distance from the current theme to the theme of reference (the Nokia tune),

n is the number of considered features,

$Reference_i$ is the value for feature i of the theme of reference,

$Current_i$ is the value for feature i of the theme being evaluated,

$Centroid_i$ is the value for feature i of the centroid of the considered bundle,

abs is the function absolute value.

The analysis was performed over a bundle of about 1000 pieces and the results (see Figure 3.1) showed that themes produced by slights variations of the Nokia tune, for example mutating the gene responsible for choosing the instrument or establishing the tempo, were measured very near to the original piece. In the middle of this aggregated dimension we observe themes evoking the Nokia theme, but with notes being split by the corresponding genetic operator (theme55) or with altered pitches and a different instrument (theme209). A little farther we find themes still preserving some structures, but with different durations, pitches and even introducing other instrument in polyphony (theme625). At the opposite side of the space there are themes whose genome have suffered so many mutations, that the resulting pieces have almost nothing in common with the Nokia tune (theme249), as the jSymbolic analysis prove and as we can clearly perceive.

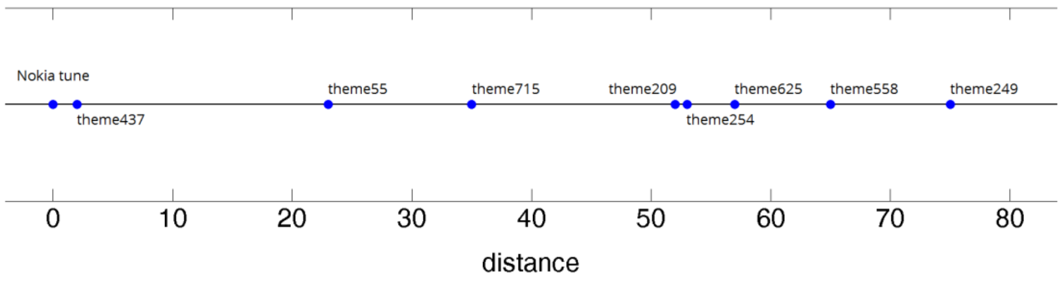


Figure 3.1. Nokia tune experiment. Illustration of the musical distance.

- Sample 3.1. Examples of the Nokia tune experiment (plain synthesis).

With the tonal system we made a similar experiment, using the same approach to assess distances, and performing the study over the phylogenetic tree described in the section Genetic engineering: SBS Logo. The original SBS Logo (node 4 in Figure 3.2) was taken as the reference; then we wanted to check whether the variations were scored higher, the farthest they were from this center. In Figure 3.2 we can see that the themes close to node 4 got low values and that, on every branch, the more mutation the themes suffered, the higher the scores were, with only a small exception, in the branch representing the crossover of a SBS variation with a theme produced by Melomics. Here the scores were likely distorted by the fact that these three themes (nodes d, 22 and 18) last some minutes, instead of some seconds like the rest of them.

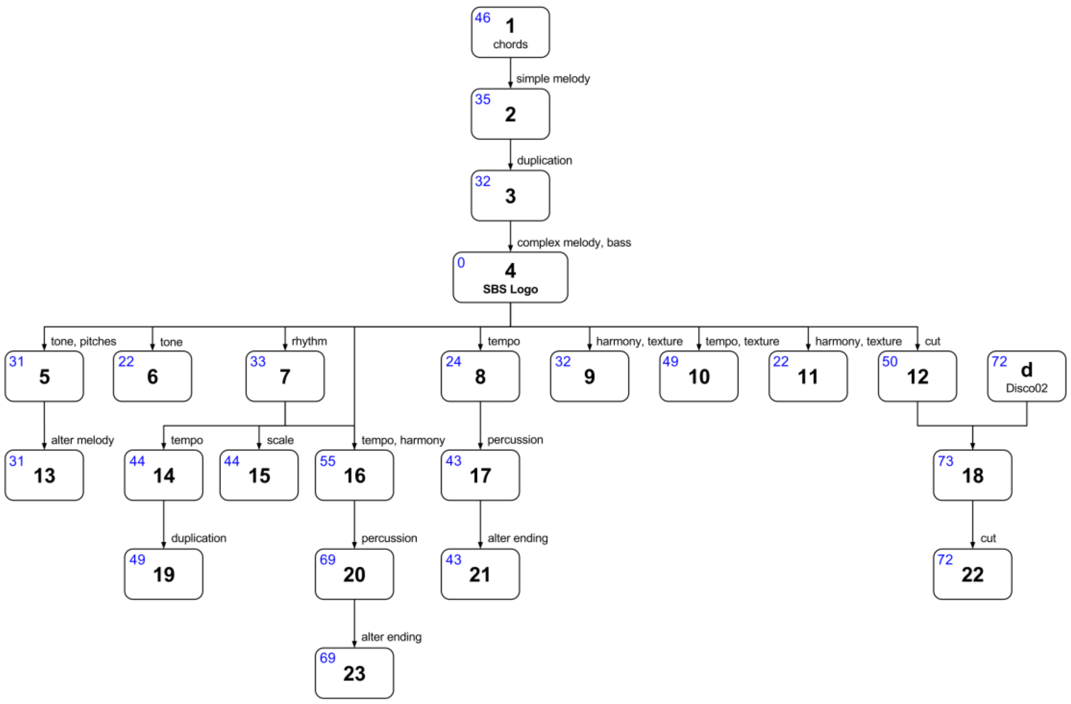


Figure 3.2. Phylogenetic tree of the SBS Logo. The main changes of a theme with respect to their parents are indicated to the right of the incoming arrow. The scores given by the analysis are placed inside each node, in blue.

- Sample 3.2. Audio of the SBS Logo experiment (plain synthesis).

3.2.1.2 Compositional analysis of the produced musical styles

Another test that we wanted to perform is to check how themes produced with the same input configuration (targeted style), are similar to each other, and how they are in relation to compositions with other configurations, and to themes created by humans. To do so, we used the same method as before to compute the centroid of each style, and then we have assessed the distances for all the considered themes, applying the next formula:

$$d_{current} = \sum_{i=1}^{n} DIndex_i$$

where

$d_{current}$ is the computed distance from the current theme to the centroid of the considered bundle,

n is the number of considered features,

$DIndex_i$ is, in a list sorted in ascending with the distances for feature i (measured from the centroid to all the considered themes), the index of the value for the current theme.

To analyze the outcomes from the atonal system, we picked a collection of 656 pieces of classical contemporary music, with eight different ensembles: bassoon, clarinet, double bass, English horn, oboe, piano, piano and violin and two pianos. After computing the centroid, we included 45 more compositions and measured the distances from all the 701 pieces to the centroid. The additional pieces were obtained from three different sources: (a) 10 pieces from the same contemporary classical substyle (not included in the process of computing the centroid); (b) 10 pieces from the style Disco02, produced by the tonal system; (b) 25 pieces from the Bodhidharma dataset[52] tagged as "Modern Classical", created by human composers.

The measurement ended with the closest composition at a distance of 13788 units to the centroid and the farthest at 26902. Considering only the initial compositions, there are a few of them close to the centroid, some others notably far from it, and most of them at a medium distance, say between 16000u and 21000u (see Figure 3.3). The new contemporary pieces from Iamus appear scattered among the originals, with a maximum distance of 20285, ranging the average zone of the style. Then, the human-created modern classical music is mostly placed inside this same average zone, suggesting that there might be a large intersection between the two styles, but on the other hand, more than a third of these pieces are located in the farthest range, since, as perceived, those particular compositions would not meet the specification of the style of reference (mainly in rhythm and harmony). Finally the style generated with the tonal system appear ranging 21023u to 26902u (the farthest theme to the centroid), meaning that not a single theme from the style Disco02 is placed in what we have considered to be the average zone of the contemporary classical substyle.

[52] http://jmir.sourceforge.net/Codaich.html

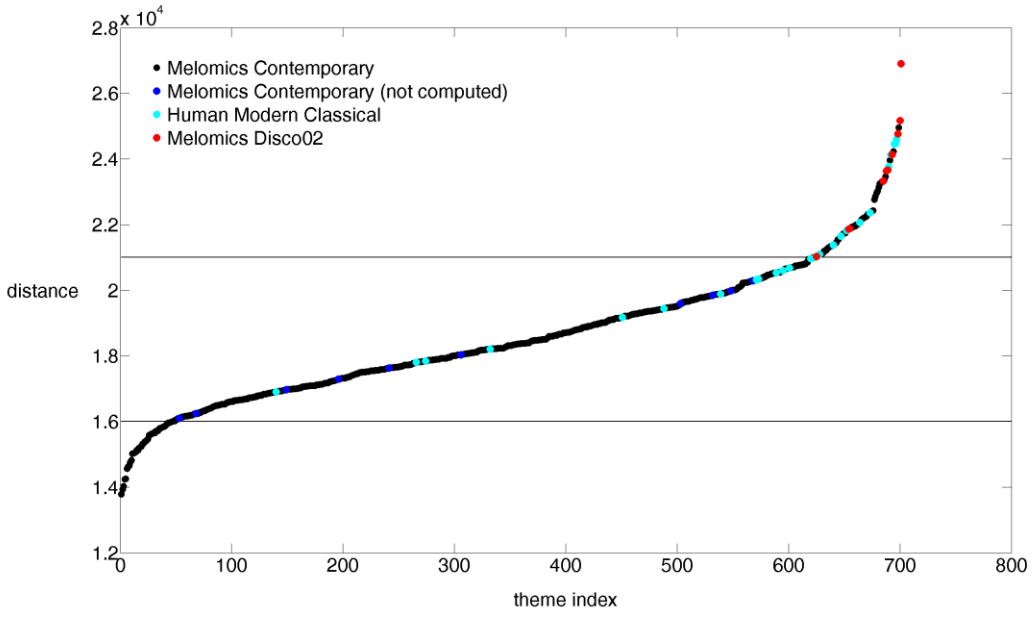


Figure 3.3. Chart of distances to the centroid of Melomics contemporary style. Distances from themes of different styles configured with Melomics and a collection of themes composed by human artists, (Bodhidharma dataset, tagged as "Modern Classical"), are shown.

To evaluate the distances between styles produced by the tonal system, which were being used for different purposes, we faced the style called DancePop (220 themes), to other two styles: Disco02 (10 themes), the style from which DancePop was derived; and DocumentarySymphonic (10 themes), a completely different style. In Figure 3.4 the distribution of distances to the DancePop centroid is shown. Once again, the DancePop themes excluded from the phase of computing the centroid (14 themes), appear sparse along the original themes. Disco themes, since they come from a style holding similarities with the reference, are ranging the whole curve too, although they are clearly scattered toward the upper side. Finally, as expected, the style DocumentarySymphonic, with typical characteristics of symphonic music, is found at the farthest distance.

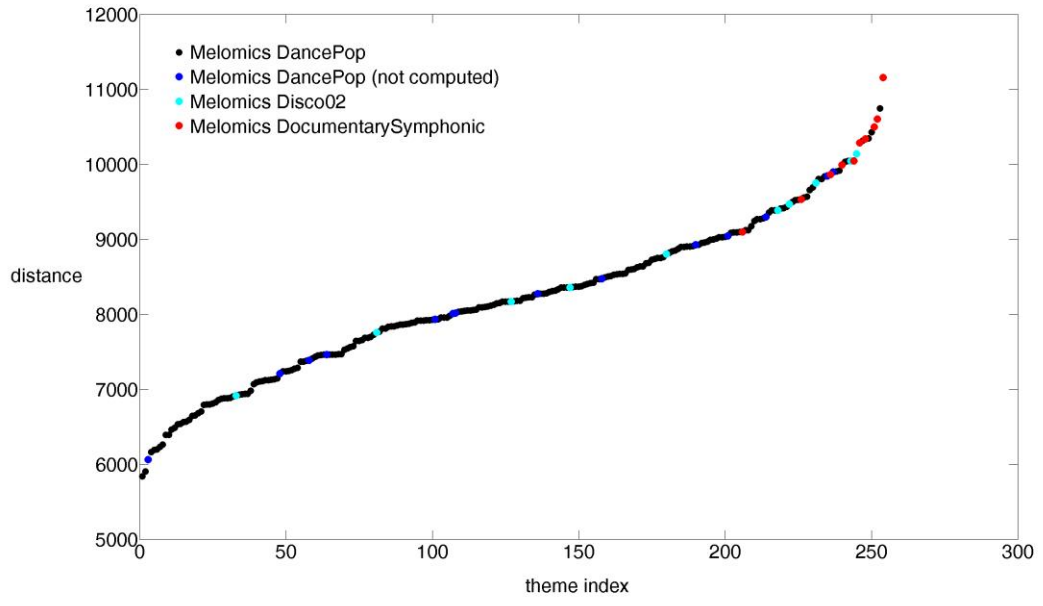


Figure 3.4. Chart of distances to the centroid of Melomics DancePop style. Distances from themes of different styles configured with Melomics tonal system are shown.

3.2.2 Melomics music among other kind of sounds

In order to put in context the different musical outputs coming from Melomics, we have made use of The Echo Nest platform. We have installed Pyechonest,[53] one of their official libraries to access the API, and in particular we have used some of their Track API Methods to upload and analyze the audio samples. Besides the expected low level data, such as tempo, loudness or key; the analyzer returns other values with a more abstract meaning, which they call *acoustic attributes*. These aggregated values, each one computed using the previously mentioned low level parameters, will range between 0.0 to 1.0, and they are: *danceability*, to describe how the audio is suitable for dancing; *energy*, a perceptual measure of how fast, loud or noisy is the track; *speechiness*, indicating the presence of spoken words; *liveness*, meaning the identification of an audience in the recording; *acousticness*, indicating if the audio was created with acoustic instruments as opposed to more artificial means; and *valence*, describing the degree of positivity or negativity conveyed by a piece. Establishing the basis for a further study of Melomics outcomes, a first exercise

[53] https://github.com/echonest/pyechonest

was to compare some different types of music made by humans, music by Melomics and other kind of sounds (natural sounds and noises), trying to find the way to distinguish the material coming each source. The samples analyzed and the abbreviations used in the figures, are listed below:

Abbreviation	**Sample name**	**Composer type**
WN	White Noise 30 seconds	Artificially generated
Tur	09 Turkey Talk	Natural sound
Rap	17 Small Rapid	Natural sound
Atl	Joan Albert Amargós - Atlantic Trio I	Human
Exe	Gustavo Diaz Jerez - Exedrae	Human
Op1	Iamus - Opus #1	Melomics
HW!	Iamus - Hello World	Melomics
Req	Mozart - Requiem	Human
Smo	Michael Jackson - Smooth Criminal	Human
Sat	Rolling Stones - Satisfaction	Human
0m6	Melomics - 0music 06	Melomics
0m10	Melomics - 0music 10	Melomics

Table 3.1. Samples analyzed and abbreviations used in the figures.

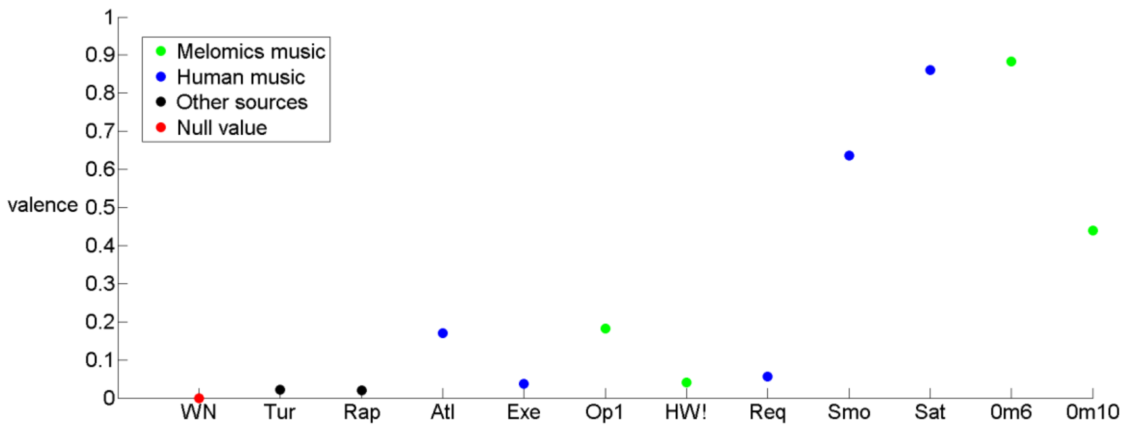


Figure 3.5. Attribute valence assessed in the 15 considered samples.

The aggregate value valence seems to provide some useful information to the purpose, since it probably makes use of the detected harmony to assess this emotional feature. In Figure 3.5, we see that the system was not even able to compute this value for the sample with white noise. The nature sounds ("Tur" and "Rap") have a notably low score, all below 0.022, a place where we could settle the threshold for a piece of audio to be considered music. And above these values, starting from let say 0.037 we find some of the contemporary classical compositions and then the rest of the pieces.

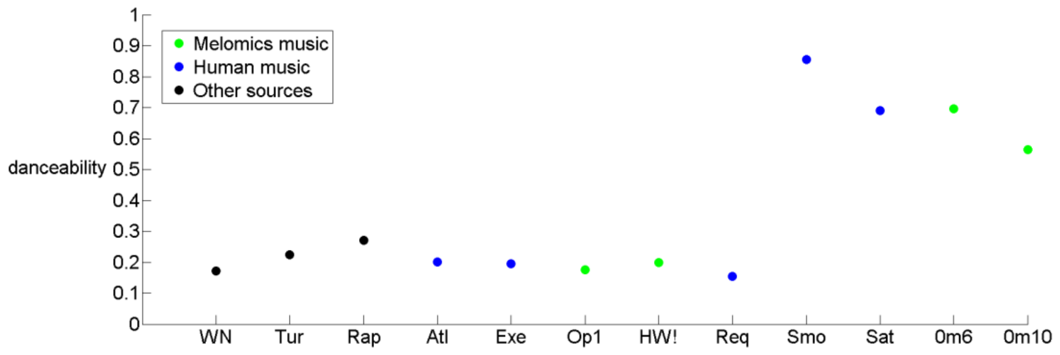


Figure 3.6. Attribute danceability assessed in the 15 considered samples.

Danceability combines tempo, rhythm, beat strength and data about regularity. Figure 3.6 shows how this attribute puts the classical music, especially the contemporary or atonal style, below the 0.3 bound, both in human and Melomics music. However, this parameter is not very useful to distinguish between this kind of music and non-musical samples, with the white noise being equally danceable as "Mozart's Requiem", for example, and "Small Rapid" natural sound more danceable than most of the classical compositions that have been analyzed. Above the 0.3 threshold there is the music that we could consider to be danceable. Perhaps combining rhythmic and harmonic information, also with an analysis of the attack and decay of sounds (to detect the occurrence of notes), might be a good way to design an automatic tool capable of telling apart what is music from what is not, and which of the compositions belong to atonal styles.

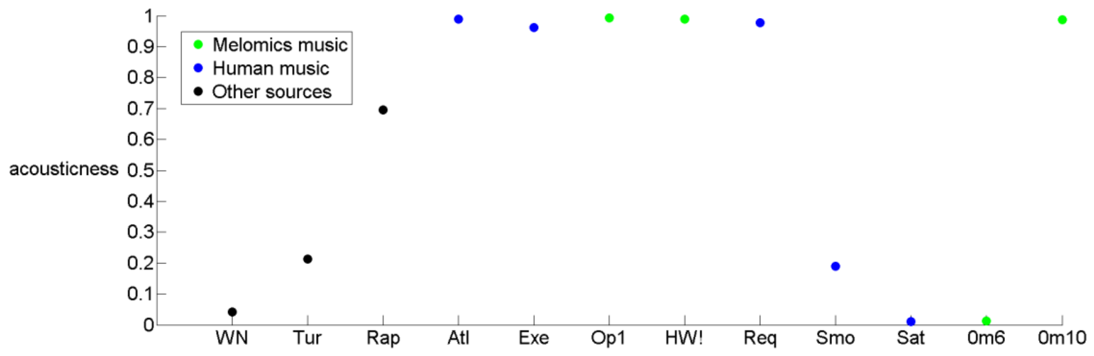


Figure 3.7. Attribute acousticness assessed in the 15 considered samples.

Another interesting indicator is acousticness; it separates very accurately the music played by orchestral instruments from those themes made with more artificial instruments. Above 0.8 we can find the first group, no matter if they were created by Melomics or a human composer, or even if the piece was synthesized by Melomics with acoustic virtual instruments or recorded in live concert, they are all in the same group. Then, below the bound of 0.1 they are the genres using electric and electronic instruments (rock, hard rock or electronic music). Between the 0.1 and 0.8 thresholds there is the music whose assessment of acousticness is unclear, probably because they include a combination of different types of instruments, or they do not include artificial instruments but they do not use orchestral ones either, or because they do include lyrics with much prominence. Natural sounds are included in the middle category and white noise is (with no surprise) assessed into the category of artificial sounds.

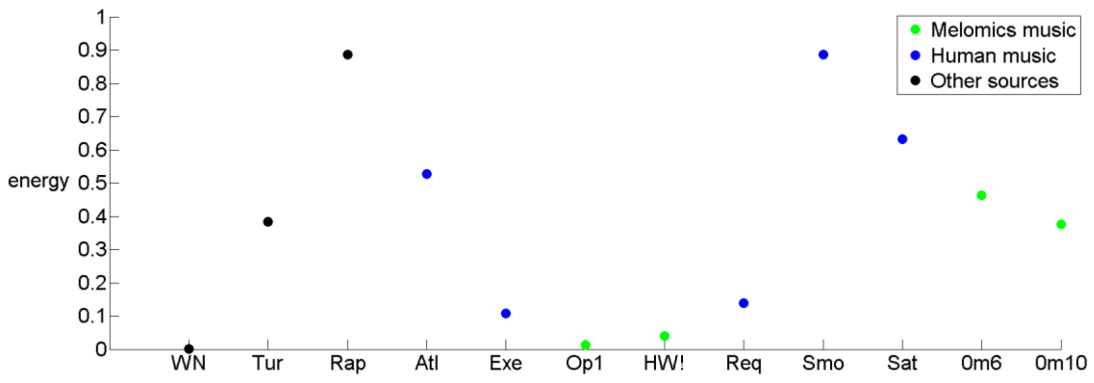


Figure 3.8. Attribute energy assessed in the 15 considered samples.

The attribute energy is computed taking into account the detected rhythm, timbres, loudness and general entropy, which is why the white noise sample is scored with almost 0, the music that we perceive as calmed has a low value (mostly classical music in this analysis), and the popular genres are located above (see Figure 3.8).

The other two of these attributes given by The Echo Nest do not provide significant information to this analysis. Speechiness resulted to be low in all compositions from Melomics, as expected, since they had no lyrics, as well as the instrumental pieces composed by human musicians. And liveness was also low for all Melomics music, even for Hello World!, which, although being an audio from an actual concert, was obtained by using professional means, with noise cancellation, etc., as it is done in a recording studio.

3.3 Perceptive study of the generated music

Automated music composition has been approached from the early times of computing. Most attempts to reach the human-level felt either in highly random works, or in creating pastiches of previously composed pieces. In 2012, the first CD of contemporary music composed by a computer in its own, original style was launched. It was also the first time that professional musicians agreed to interpret computer-composed pieces (LSO recorded two full-orchestra pieces, and other first-shelf artists performed the chamber orchestra works). In a time that is seeing the revival (in big data, HCI, robotics) and also the potential threat of AI, the question of whether computer-composers were attaining (or would attain in the near future) human expressiveness blows in the air.

While measuring the performance of an assembly robot in a production line is straightforward, assessing the quality of other AI systems, particularly those involving simulated cognition or creativity, has always been controversial. To date, the Turing Test (TT), Alan Turing's proposed imitation game that intended to answer the question "Can a machine think?" (Turing, 1950), is still considered the best way to determine if an artificial system involves some sort of thought. Even if Turing's proposal was epistemic, rather than ontic (and more heuristic than demonstrative) (Harnad, 2000), it is probably the best approach to date to this problem. That is probably the reason that a wide range of variants have been proposed (as a generalization of the Turing Test) to measure the success of AI systems in related areas. Other tests have proposed, and some of them understand the term “creative” in a very restrictive sense, like the Lovelace Test (LT), which demand full creativeness on the machine under study, which is hard to meet (and verify) for an artificial system. Basically, the LT proposes that a system (say, a program) is creative if its work cannot be traced back by the author of the system (the programmer). An interesting, and more practical, variant is the Musical Directive Toy Test (MDtT) (Ariza, 2009), which allows the interrogator as to control the experiment by providing musical directive to the composing systems under study (human and computer). This moves in the direction of making the test more interactive, although the times in composition differ from those in a spoken conversation. On the other hand, the Musical Output Toy Test (MOtT) focuses more on the results of the composition, no matter how the composers were commissioned. In both musical tests, it must be taken into account that music is abstract in nature, so there is no underlying grammar comparable to that of natural language. For that reason, MDtT and MOtT, remove the essential component of natural language discourse, while keeping the blind indistinguishability of the TT. As such, both tests must be considered surveys of musical judgments, not a measure of thought or intelligence.

3.3.1 Experiment to evaluate Melomics composing capability

While research in generative music systems shows a lack of experimental methodology (Papadopoulos & Wiggins, 1999), and typically no evaluation of the output by real experts, the raise of affective algorithmic composition (Williams, et al., 2015) has promoted new strategies of evaluating the results of automatic composition systems, as compared to human produced music. The lattermost study was described by (Delgado, Fajardo, & Molina-Solana, 2009), which considers a compositional system based on expert systems, with information about emotions as input; and then they evaluate to what extent did the listeners matched the original emotions. In this paper, we describe a MDtT designed as to assess Melomics’ compositional and synthesis capabilities. The experiment involved 251 participants, half of them professional musicians and half non-musicians, who have reported on the emotions that were elicited while listening to two different pieces of music, and then to identify the one composed by Melomics (the other was composed by a musician). As we will see, the main contribution here is the rigor of the study, both at the design of the experiment, and at the significance of the results.

3.3.1.1 Experiment design

The ultimate goal of this experiment is to measure the valence of computer-composition and computer-interpretation, as compared to their human counterparts. This section describes in detail the methodology of the experiment.

Two different music pieces were created, one by a human compositor, and another by Melomics109, with the same specifications: the style was a guitar ballad; the instruments to be used were piano, electric piano, bass guitar and electric guitar; the bar was 4/4; the BPM was fixed to 90; the duration was 2 minutes; the structure had to be A B A´; and it has to be in major scale. The final audio files were obtained in MP3 format with a constant bitrate of 128kbps, and shortened to a duration of 1’42”, such that the beginning and ending of the pieces (around 9” each) were removed, since they typically render constant composing patterns in popular music.

After having being composed, both pieces (the human- and the computer-composed ballads) were interpreted by a human musician and by the computer. The human interpretation was done upon the scores, while the computer automatically synthesized and mixed a MIDI representation of both pieces. This yielded four combinations, and a two-minute excerpt from natural sounds (The sounds of Nature Collection (Jungle River, Jungle birdsong and Showers), 2010) combined with animal sounds (Animals/Insects, n.d.) was added to the set, in order to measure the response to non-musical sound. This is how the five pieces were labelled:

		composed		
		human	computer	nature
interpreted	human	HH	CH	-
	computer	HC	CC	-
	nature	-	-	NS

Table 3.2. Labeling of pieces.

where HH stands for human-composed and interpreted, CC for computer-composed and synthesized, HC for human-composed and computer-synthesized, CH for computer-composed and human-interpreted, and NS for natural sounds.

The test was implemented in a way that each subject was informed that she is undergoing an experiment in music psychology, not mentioning that it involves computer-composed music, so as not to bias the results (Moffat & Kelly, 2006). Then, the subject is assigned a group, and listens to five pieces, in two phases, as shown in Table 3.2. The distribution is such that the subject listened to both pieces as rendered by only one interpreter, making the decision independent of the quality of the execution.

	Group A	Group B
Phase I	HH / CH / NS	CC / HC / NS
Phase II	HH / CH	CC / HC

Table 3.3. Groups and phases of the experiment.

During Phase I, each subject in group A listened to the three pieces randomly ordered. HH and CH have been composed by a human and by the computer, respectively, and both are interpreted by a human musician. Subjects assigned to group B proceed similarly; they listened to the same compositions, in this case synthesized by a computer. Finally, both groups listened also to natural sounds. After having listened to each piece, the subject is asked whether that excerpt could be considered music, and what mental representation or emotional states it has evoked in the listener. This last question has to be answered in an open way, in the sense that the subject was not given a list of alternative answers.

In Phase II, the same musical pieces are played (not the natural sounds), but this time the subject is prompted with the question of whether the piece has been composed by a human or by a computer. As previously stated, it is important that the identification of the author is made in the second phase, in a way that the subjects can evaluate the valence of the music in Phase I without the bias of knowing that part of the music is composed by a computer.

- Sample 3.3. Bundle of themes used in the perceptive study.

3.3.1.2 Experiment results

The experiment was performed in two facilities: the Museum for Interactive Music of Malaga, and the Music Conservatory of Malaga. Recruiting of subjects was promoted with a banner for the museum visitors, and through an internal call among students and educators at the conservatory.

Selected participants ranged in age from 20 to 60 years, and the answers were processed differently according to their music expertise: subjects with five or more years of training (labelled as "musicians"), and subjects with less or no training (labelled as "non-musicians"). Musicians are assumed to process music in a much more elaborated way and possess a wider knowledge on music structure, so they are expected to outperform non-musicians in a music classification task. The final sample resulted in a group of 149 musicians, and another group of 102 non-musicians (251 subjects overall).

Phase I	group A	group B	group A	group B	group A	group B
Is this music?	**HH**	**HC**	**HH**	**HC**	**HH**	**HC**
yes	98.51	96.58	98.88	100.00	97.78	92.98
no	1.49	3.42	1.12	0.00	2.22	7.02
Does it elicit mental representations?						
yes	70.15	62.39	68.54	60.00	73.33	64.91
no	29.85	37.61	31.46	40.00	26.67	35.09
Does it elicit emotional states?						
yes	82.09	82.05	84.27	81.67	77.78	82.46
no	17.91	17.95	15.73	18.33	22.22	17.54
Is this music?	**CH**	**CC**	**CH**	**CC**	**CH**	**CC**
yes	97.76	99.15	97.75	100.00	97.78	98.25
no	2.24	0.85	2.25	0.00	2.22	1.75
Does it elicit mental representations?						
yes	63.43	55.56	66.29	51.67	57.78	59.65
no	36.57	44.44	33.71	48.33	42.22	40.35
Does it elicit emotional states?						
yes	87.31	80.34	89.89	76.67	82.22	84.48
no	12.69	19.66	10.11	23.33	17.78	15.52
Is this music?	**NS**	**NS**	**NS**	**NS**	**NS**	**NS**
yes	36.57	38.46	41.57	41.67	26.67	35.09
no	63.43	61.54	58.43	58.33	73.33	64.91
Does it elicit mental representations?						
yes	91.79	95.73	94.38	95.00	86.67	96.49
no	8.21	4.27	5.62	5.00	13.33	3.51
Does it elicit emotional states?						
yes	83.58	84.62	84.27	81.67	82.22	87.72
no	16.42	15.38	15.73	18.33	17.78	12.28
Phase II	group A	group B	group A	group B	group A	group B
Human- or computer-composed?	**HH**	**HC**	**HH**	**HC**	**HH**	**HC**
computer	48.51	49.57	49.44	51.67	46.67	47.37
human	51.49	50.43	50.56	48.33	53.33	52.63
	CH	**CC**	**CH**	**CC**	**CH**	**CC**
computer	45.52	56.41	51.69	65.00	33.33	47.37
human	54.48	43.59	48.31	35.00	66.67	52.63

Table 3.4. Results of the survey. All questions are referred to the piece that the subjects have just listened to, and the results are given in percentage.

Data analysis was performed with a fourfold contingency table (two by two), and the differences between groups were evaluated by the Chi-square test with correction for continuity. Only in those cases where the expected frequency was lower than 5, the Fisher test was used. The level of significance was established at p<0.05. In the analysis of Phase II data, it was also evaluated the sensitivity (i.e. the capacity to correctly identify the pieces composed by the computer) and the specificity (i.e. classification of pieces composed by humans: the capacity to identify that it is not composed by a human composer), for both musicians and non-musicians, at a 95% confidence level, see Table 3.4.

	musicians			non-musicians		
	value	CI (95%)		value	CI (95%)	
sensibility (%)	57.05	48.76	65.33	41.18	31.14	51.22
specificity (%)	49.66	41.30	58.03	52.94	42.76	63.12

Table 3.5. Sensitivity and specificity.

As a result of this analysis, no differences were identified as for who composed or interpreted the music (be it the human or the computer), and this lack of discrimination was measured both for professional musicians and non-musicians. Remarkably, natural sounds showed to be as good as music at eliciting emotions, and even better at generating mental representations.

With the advent of the new trend in generative music, based on defining original styles, rather than imitating human authors, artificial creativity has reached higher standards in music generation. Therefore, it is more necessary than ever to assess the quality of the compositional structure and the performance of these computer-musicians, as to be able to compare it with that of human experts.

The Turing Test was designed for identifying thought traces in computer processing, and its interactive nature makes difficult the adaptation to a musical version (where the interrogator has to identify the computer-generated music). Nevertheless, the underlying principle of Turing's approach remains as a valid inspiration for new tests that establish how human computer-composers have got. The experiment described above illustrates this point by defining a controlled and rigorous trial, designed as to check the music valence of computer-composed music, with respect to a similar effort performed by a human composer. Also, the experiment shows that, when there is no interpretation bias, professional musicians and non-musicians cannot distinguish the nature of the composition. The size of the sample (251 subjects) is big enough for the results to be considered representative. The fact that about a half of it is made of professional musicians is an indicator of the robustness of the conclusions (even if most of the standard population —the potential market for computer-composed music— has less than five years of musical training). This is, of course, a first result in this line of quality assessment. The model can be extended to different musical styles, given that the level of synthetization (in that particular style) is good enough as to be considered quality-music by the subjects.

Initially, this experiment was thought to prove that Melomics' music can be really considered as human music (in the sense that it cannot be differentiated from a human composition), and that it generates a similar effect in the listeners. The results of the study confirm this hypothesis and pave the way for using Melomics' compositions as true music. One of the main uses of this technology is as music medicine in healthcare (Requena, et al., 2014). The potential of generative music that is tested as human music is vast, since it allows to create tailored music (specific genre, design, and parameters fitting). In this sense, the flexibility of Melomics is particularly adequate to adapt the music to the therapeutic needs and goals, also considering the patient's personal preferences. Subscribed to this line of research, Melomics is being tested in clinical trials to prove its efficacy in different clinical conditions: chronic and acute pain, anxiety, sleeping disorders, and surgical interventions.

3.3.2 Additional evaluations

Apart from the described experiment, Melomics music has been tested in several other less controlled contexts:

On May 2015, a professor of the Conservatory of Malaga performed *Just out of curiosity*, a newly generated composition for clarinet by Melomics, in front of many students. Additionally, another composition from Melomics was played back to the audience. Following this, and not having mentioning the source of the music, the professor asked them to highlight any aspect of the compositions, for instance whether they could identify the historical context, name a possible author, or just give an opinion on the pieces. After a number of interventions, the students were told who the actual composer was. The general conclusion was that they would not expect these compositions to come from a computer, with no human intervention or being fed with previous pieces.

- Sample 3.4. Compositions played at the Conservatory of Malaga.

Another test with Melomics music was carried out during the SDF2015 event. It was a version of the experiment described in Section 3.3.1, with a similar purpose, to explore to what extent do computer-composition and computer-interpretation have the same valence than their human counterparts. Subjects also ranged in age 20 to 60 years, with a population of 150 musicians and 350 non-musicians, and they were asked the same questions about the same music and natural sounds as before. The experiment was performed with the help of the platform OPENSURVEY™,[54] through a web-based questionnaire. Results showed that Melomics music elicited the same kind of feelings and mental representations, different than that evoked by natural sounds. Also, success rates when trying to distinguish human-composed from computer-composed music was around 50%. Only in the case of pieces synthesized by the computer, professional musicians had a slightly perceptible success in the identification.

Regarding the methodology that has been followed in all these assessment, it is worth mentioning that, as supported by neuroscientific studies (Steinbeis & Koelsch, 2009), critiques to computer-made compositions are suitable to be affected by anti-computer prejudice, if knowing in advance the non-human nature of the author. To illustrate this fact, we present two examples of critiques to Hello World!. In one case the critic (Service) is aware that the composer is Iamus, while in the other (Russell) is not.

Tom Service, music critic for The Guardian, in reviewing a performance of Hello World! by human musicians, noted that the piece "...sounds like it's slavishly manipulating pitch cells to generate melodies that have a kind of superficial coherence and relationship to one another, with all the dryness and greyness

[54] http://www.opensurvey.co.kr

that suggests, despite the expressive commitment of the three performers" (Service, 2012).

Peter Russell, musicologist, was asked to review Hello World! for the BBC, based on a video of the life premiere, but he was not given any information about the composer. In his critique, Russell writes "on listening to this delightful piece of chamber music I could not bring myself to say that it would probably be more satisfying to read the score than listen to it. In fact after repeated hearings, I came to like it" (Smith, 2013) (Ball, 2014).

3.4 Summary and discussion

In this chapter we have analyzed the music created by Melomics. We have done this by reviewing a wide range of its compositional styles and following different approaches. As a summary, by analytical methods we have concluded that the similarity between genomes is tightly connected to the similarity measured in the corresponding musical phenotypes; and we have seen that Melomics is able to produce music in an arguably wide variety of styles, being close to their human counterparts and distant to other kind of sounds. A further analytical study is intended to be performed, using additional MIR tools, an even wider selection of features and a more advanced analysis of them, including clustering techniques, especially to study the differences between styles. The second half of the chapter has been focused on a more empirical analysis, based on human subjective assessments. The results seemed to point into a similar conclusion: Melomics music is perceived in the same way that the human-composed music, and clearly different from other types of sounds, as long as the critic is not biased by the fact of knowing the computer nature of the composer.

Chapter 4

Adaptive music composition

In this chapter we introduce a way of creating and providing music, capable of responding dynamically to the circumstances of a considered environment, possibly including an assessment on the current listener's state. In Section 4.1 we explain how can we benefit from automated composition and present our proposal of adaptive music; in Section 4.2 we summarize some previous attempts on the area; in Section 4.3 we define an specific implementation of the adaptive music system; and in the Section 4.4, we show and discuss the different implemented applications based on it, designed for different scenarios, and capable of producing sentient responses through music.

4.1 Motivation

Designing a computational system capable of creating music, in the same way an expert human composer could do, as a research target, is ambitious and compelling enough by itself. However, such an innovative tool can cause the appearance of new applications and lines of work, which may have not even been considered, due to the unfeasibility of some tasks to be carried out by humans; hence we have approached some of them too. As an example: probably anyone would have wanted sometime to listen to a precise type of music, and not just the music created by others that more or less fit their expectations. However, such thing has not occurred, because usually people lack of the sufficient knowledge to create the music by themselves, and there are no composers enough in the world that could satisfy such demand; besides, it would not be an ideal job for a musician, but is indeed well suited for a computer (able to compose). Therefore, in this case, it is not as much a human activity being replaced by a computer, but the computer filling a place that is not designed for people. Music tasks that imply just satisfying a bunch of specification would be no longer an obligation of the composers, in the same way it occurred with the invention of photography, in regard to the tasks of producing portraits or capturing scenes; moreover, it should bring new ways of doing art.

The main advantages identified of a computer composer are: (1) producing one composition or a bunch of them faster; (2) creating themes without deviations from the specifications; (3) creating variations of compositions, by modifying simple elements in a precise way, or through major transformations (affecting the structural basis, harmony, texture, etc.). As commented in previous sections, we have experimented with all these possibilities, but as a clear line of applications resulting from this work, we have focused our main efforts on *adaptive music*. This consists of making music in real time, by handling the musical parameters in order to respond to certain values identified in the environment. A simple example to illustrate this idea will be taking the stage of a disco, with the average speed of movement detected as the analyzed parameter. The music to provide could be adjusted, by bending the tempo, volume or textural density, for example, to be proportional to the detected movement of people, being the music a reflection of the energy perceived at the stage.

4.2 Related work

Audio being adjusted in real time in order to answer to certain input parameters can be found in some fields of application. The most widespread case might be its use in computer games. Practically since the beginning, playing music according to the situation presented to the player was seen as an important part of the dynamic. Lately in the 90s, when most video games were made using 3D engines and were aimed to provide an immersive experience, music also achieved a greater importance, with titles like Silent Hill or Resident Evil, which made use of specific music to create the desired fearful atmosphere, or the stealth game Metal Gear Solid, where music indicated a delicate moment or told the player that something was going to happen and needed to prepare for a fight. Experimenting with adaptive audio, not pre-synthesized, but created in real time according to certain rules and inputs, in electronic entertainment and other interactive settings, has been a tendency in recent times. This matter has also been studied in science, usually referred to as non-linear music (Buttram, 2003) or procedural audio (Garner, Grimshaw, & Nabi, 2010); an example technique is given by (Wooller & Brown, 2005), which applies Markov chains to generate musical transitions between two different pieces.

Also, regarding inventions to exploit this idea, we have found the patent (Patent No. DE3338649 A1, 1985) consisting of a device for emitting acoustic signals (sounds, music and/or speech) in order to promote sleep, with the signals controlled by a physiological parameter registered in the individual; the patent (Patent No. US20090149699 A1, 2009) describes a similar system but with a more complex analysis of the physiological parameter, characterizing the sleep phases in the subject, to act in consequence; and patent (Patent No. WO2003018097 A1, 2003), a system that produces acoustic signals and is addressed to treat stress pathologies.

A more recent application of adaptive music and nearer to the idea that we are trying to expose is given by (Morreale, Masu, & De Angeli, 2013). This setting includes an algorithm that is designed to produce MIDI output in an interactive context. The inputs considered are based on the emotions detected in certain audience, for example extracting information from the movements of people while dancing (Morreale, De Angeli, Masu, Rota, & Conci, 2014), so the algorithm can adjust some global parameters like volume or tempo and some specific features like amount of dissonance and density of notes.

As a summary of the cases explored, we have observed that there are either (a) systems based on pieces pre-composed by human artists in the traditional way, which are played sequentially, selected with certain criteria and possibly using transition effects (typically crossfade); or (b) systems for live generation of audio signals, far from being real music compositions. Our contribution to this area is a method that is capable of providing genuine music to the audience, responding in real time with the possibility of treating any musical parameter at any level of abstraction. The rhythmic evolution, the harmonic curves, the instrumental density, the relationships between instrumental roles or the compositional structure are some of the parameters that can be handled.

4.3 Feedback-based music: The hypersong

As shown before, music coming from Melomics can exhibit similar properties than the music produced by a human musician in the same style. In this section we will propose an implementation of an adaptive musical system, taking advantage of this fact. To illustrate it, we will use a case example consisting of a mechanism able to characterize the amount of nervousness of an individual, through monitoring the heart rate, for example, and to play music with the aim of reducing that perceived anxiety. The system could be structured in the next elements:

- A component to measure and characterize the environment (in the example, the heart rate sensor and the function that summarizes the provided data in three levels, for instance: anxious, active and relaxed).
- A mechanism that matches the state perceived in the environment to the appropriate type of music to be played.
- A system capable of producing the convenient flux of music.

A way of implementing this behavior with Melomics could be based on playing its compositions while at the same time the environment is being evaluated iteratively, so that when a different version of the music has to be played, the proper parameters are altered in the genome and the composition is regenerated with this new configuration, and then played instead of the previous version of the theme.

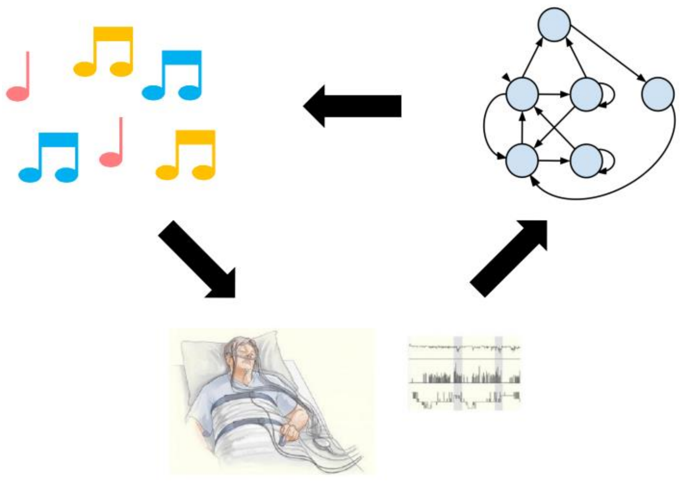


Figure 4.1. Feedback-based music system. It comprises characterization of the user or the environment, selection of music and production of music.

In practice, this way has technical limitations. First, the system will typically be working in a small laptop or even in a mobile device. It would not be difficult to optimize the composing and synthesis processes to be executed in this hardware faster than the music is played in real time. Nonetheless these operations are very computing expensive, causing the devices to work in a higher temperature range than desired and dramatically shortening the battery life, forcing us to use a charger, which is not feasible in some scenarios. Another consideration is that, in order to synthesize a piece of music with the expected level of quality, a professional library of virtual instruments is required. This tool would be stored in the devices running the adaptive system, occupying a huge amount of storage. Even ignoring those problems and supposing we can count on a high-end computer to execute the mentioned workflow, the method can easily suffer from failure. If the system has to compose and synthesize (expensive processes) in real time, any occurrence of an external event may cause a critical delay, not being able to produce the following segment in time, resulting in an interruption of the musical flux.

To solve these limitations, we propose a variation of the described system: For an application, we will settle in advance the different required versions of the music and we will pre-generate all the audio files. The versions may vary in a number of features, but the general structure will be preserved among them. They will be fragmented in the time dimension, making the cuts between different structural entities (see 2.2.3.1 Musical styles), so that the different versions may have a different duration, but the same amount of fragments. As a result, for each composition, we will produce a two-dimensional musical entity that we called a *hypersong*, containing also some meta-information, as the usual music produced with Melomics.

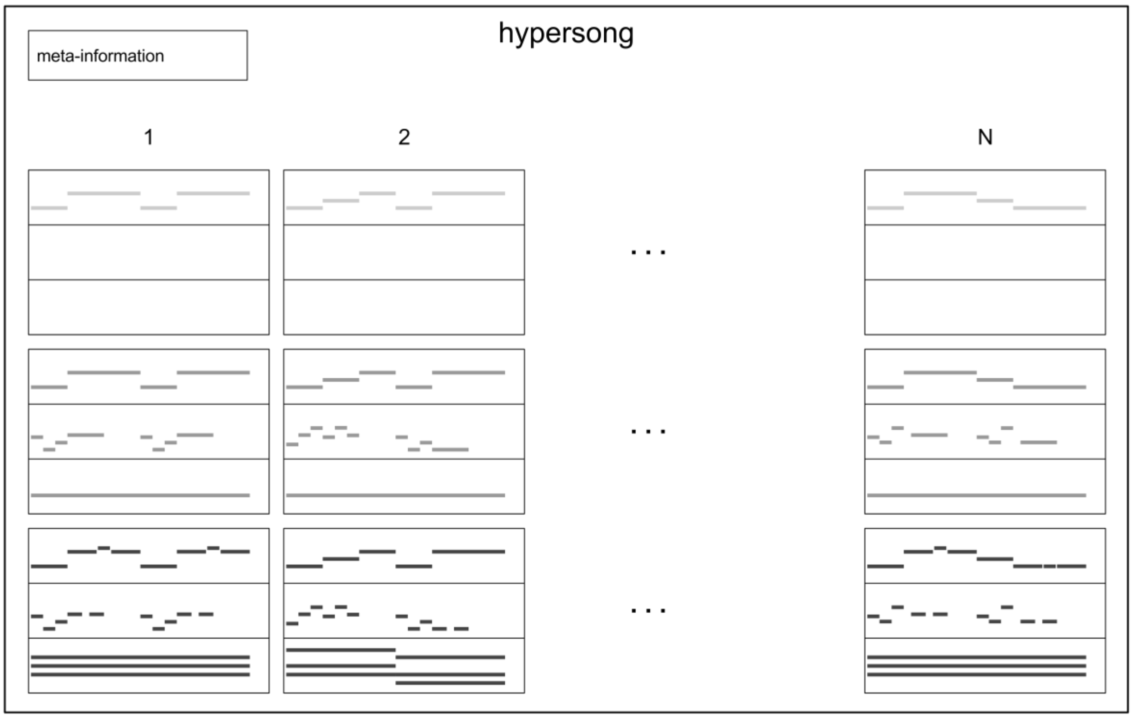


Figure 4.2. Illustration of the hypersong. Each of the three versions in the example (represented vertically), is divided in N fragments in the time (horizontally). The versions comprise from one to three instruments, as indicated by the vertical divisions inside the fragments. The different fragments are made of different musical content, although each instrument is supposed to behave in a similar way along the time. The difference between the versions are: from the first to the second one, the addition of two instruments, all playing at a higher volume (represented by the brightness of the lines); and from the second to the third version, the increment of complexity and volume in the three performing instruments.

By using this approach, there are some considerations that shall be discussed. First, this method may seem more computationally expensive, since we have to produce and store all the versions for each hypersong; nevertheless, in the long term is more efficient, since most fragment will be played eventually, so the hypersong is computed just once at the beginning, instead of having to produce each fragment every time that is needed. On the other hand, the hypersong method consumes more storage indeed; however, we can minimize this problem by producing and storing the songs in a remote server, liberating the mobile device from this task, just having to download the proper fragments to play. This may imply some additional problems, like network failures or consuming too much of the data plan if this app is used through the cell carrier network instead of a WIFI connection. But once again, there are some solutions to these problems, like caching fragments according to the user profile and the historic data of characterization. Moreover, the designed system based on hypersongs has some clear advantages over the commented primary solution: (a) Very little computer power is required from the client devices, which is one of the main goals. The remote devices almost only have to run a quite simple mechanism of streaming playback, instead of being in charge of deciding the parameters to make the music evolve, developing and synthesizing the piece of music, and playing the audio. (b) Both the software in charge of generating music and the software to decide how music should evolve in response to the characterized state, run in a remote server, which eases to preserve the privacy of the algorithms if so desired.

To make the system capable of producing the hypersongs, we had to introduce some modifications. We coded new scripts that enabled to produce a musical theme and, at the same time, its specified variants, changing some parameters but preserving the essential parts of it. Another task was to design the way the versions are cut into fragments. The first consideration is that these cuts cannot be based on time marks, mainly because different versions of the same hypersong might have different durations, because of different tempos or rhythmic patterns. Thus, cut were based on compositional information, which did not imply a major change in the implementation, since the system already saved this meta-information. Another issue was to decide where exactly should we settle the breakpoints to cut, because the flux of music provided by the adaptive system should be perceived by the listener as a continuous evolving composition; hence, the cutting points were decided to be between two or more compositional ideas (see 2.2.3.1 Musical styles), avoiding to break any of them. An additional time of silence was needed to be added at the end of each fragment, to assure the proper decay of the notes played by the virtual instruments. After running some tests, we decided to include two measures of silence in most of the cases. Usually, we produce fragments with the proportion

2/3 of actual composition and 1/3 of rest for sound decay, providing a sufficient speed of response and an acceptable 1/3 of storage overload.

- Sample 4.1. Same temporal fragment of a hypersong in its three levels.
- Sample 4.2. Evolution of a hypersong through its three levels.

As a result we developed a system for adaptive music where the elements mentioned at the beginning of this section are identified as follows: (a) a software to identify states in real time, whose general description will be outlined in the next section and some specific designs will be given in Section 4.4, for each application; (b) the mechanism to map the states to each level or version of the hypersongs, which will be explained in Section 4.3.2; and (c) the system in charge of producing the flux of music, integrated by the hypersongs generated with Melomics, and being played by the client application, according to the directions given by the second component.

4.3.1 Characterization of states

An essential element of the adaptive system is the one in charge of characterizing the states, according to which the music should be adapted. This component can be described through a global architecture, shared by all the applications: (a) one or more sensors take measures from the environment; (b) processing functions are applied on each of the registered signals; (c) information of a higher level of abstraction is obtained after an specific analysis of the signals; (d) an input state is computed and provided to the automaton, as a result of the combined information that has been gathered. As will be shown in Section 4.4, some of this modules of characterization consist of quite simple functions, like thresholding of a signal coming from a single sensor, some others include more complex and computationally expensive operations, like identifying rhythms of movement using the 3-axis accelerometer signal, and some others even combine information from sensors of very diverse nature.

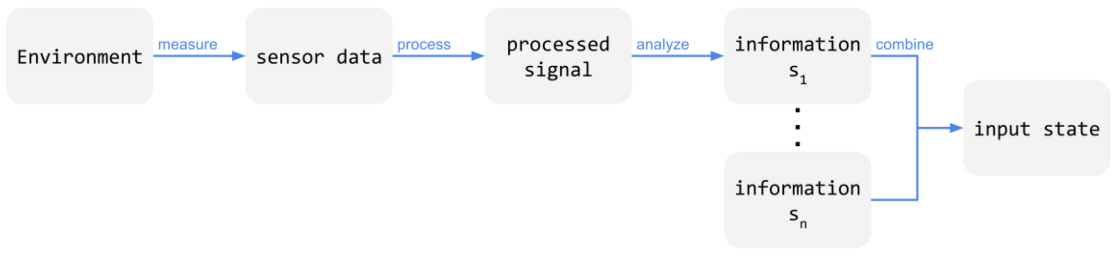


Figure 4.3. Process of characterization.

4.3.2 Control of compositional paths by DFA

4.3.2.1 A model based on Moore Machine

Once the present state is identified, we need a mechanism capable of using this information and deciding the right music to play, depending on the intended purpose of the application. In order to implement this functionality, giving support in a standardized way to all the applications based on adaptive music, we have propose a behavior based on the Moore Machine, a deterministic finite automaton (DFA), plus some modifications to adapt it to our needs. To the standard Moore model, we add (1) a *time counter*, reset after every transition; (2) an *insensitivity time* for each state, to force to hold for a certain period of time, before letting any new transition to occur; and (3) a *null-input timeout* parameter for each state, to force a default transition from the current state to another, if no input is received by the state machine during this period. The model can be defined as an 8-tuple $(Q, \Sigma, \Delta, q_0, F, T, G, M)$ where:

Q is the set of states

Σ is the input alphabet including the null value

Δ is the transition function, $\Delta\colon \mathrm{Q} \times \Sigma \times \mathrm{T} \rightarrow \mathrm{Q} \times \mathrm{T}$, such that:

$\Delta(q_i, \Sigma, \mathrm{t}) = (q_i, \mathrm{t}+1), \text{if } \mathrm{t} < I_i,$ being I_i the insensitivity time given for the state q_i

$\Delta(q_i, \phi, \mathrm{t}) = (q_i, \mathrm{t}+1), \text{if } \mathrm{t} < \lambda_i,$ being λ_i the given null-input timeout for the state q_i

$\Delta(q_i, \phi, \mathrm{t}) = (q_{\lambda i}, 0), \text{if } \mathrm{t} \geq \lambda_i$, being $\lambda_i$the null-input timeout for the state q_i and $q_{\lambda i}$ the null-input timeout transition state defined for q_i

$\Delta(q_i, \Sigma_k, \mathrm{t}) = (\delta(q_i), 0), \text{otherwise}$, with $\delta(q_i)$ the state defined by a traditional Moore Machine transition function $\delta: Q \times \Sigma \to Q$

with $I_i < \lambda_i \ \forall i$ and $t, I_i, \lambda_i \in T$

q_0 is the initial state, $q_0 \in Q$

F is the set of final states, $F \subseteq Q$

T is the set of discrete instants of time, $T \subseteq \mathbb{N}$

G is the output function $G: Q \to M$

M is the output alphabet

In Section 4.4, some concrete instantiations of this model will be detailed.

As a summary, the system in charge of controlling the adaptive playback will be compounded by this mechanism based on finite state machines; a small logic to keep the count over the playlist of hypersongs, including the theme and fragment currently being played; and a media player for hypersongs that is formed by two traditional media players that intervene in an alternate fashion: one plays the current fragment, while the other is preparing the next fragment, starting to play right when the first one is over the decay section of the previous fragment. When the first player is finished, prepares the next fragment and the process starts over with the roles having been interchanged.

To ease the process of designing a behavior based on this DFA model, we made use of JFLAP,[55] an open source free software developed at Duke University. The tool was modified to allow us to create and edit our particular kind of automatons. These designs are stored in XML files, which can be easily converted to JSON and interpreted by the software in charge of controlling the adaptive system.

[55] http://www.jflap.org/

4.3.2.2 Implementation through web services

The implementation of the commented mechanism has evolved from the first rapid prototype written in MATLAB, where the transition functions for each application were every time hardcoded in a separate ".m" file, to a mobile oriented platform, where the procedure of creating state machines was standardized and simplified, making use of a graphic tool that produces a configuration file that can be saved and exported.

```
{
   "_id" : <DFA_UUID>,
   "model" : {
      "outputs" : <List_with_the_output_for_each_state>,
      "numMetaThemes" : <List_of_hypersong_indexes_of_the_metaThemePrefix_class>,
      "metaThemePrefix" : <Hypersong_class_to_be_used>,
      "initialState" : <Initial_state>,
      "delay" : <List_with_the_insensitivity_time_for_each_state>,
      "finalState" : <Final_state>,
      "table" : <Transition_function_based_on_a_list_of_2-tuples_for_each_state>,
      "introTheme" : <Possible_initial_theme_to_be_played_before_the_DFA_starts>,
      "lambdas" : <NullIinput_timeout_based_on_a_list_of_a_2-tupla_for_each_state>
   },
   "name" : <DFA_NAME>
}
```

Figure 4.4. Melomics DFA template for adaptive music applications.

```
{
   "_id" : "88e5f38b-123c-447d-99af-c217d0c15213",
   "model" : {
      "outputs" : [ 0, 1, 2, null ],
      "numMetaThemes" : [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 ],
      "metaThemePrefix" : "painnightalter",
      "initialState" : 0,
      "delay" : [ 60, 0, 0, 0 ],
      "finalState" : 3,
      "table" : [ [ [ 0, 0 ] ], [ [ 1, 0 ] ], [ [ 2, 1 ] ], [ [ 3, 3 ] ] ],
      "introTheme" : "",
      "lambdas" : [ [ 60, 1 ], [ 600, 2 ], [ 600, 3 ], null ]
   },
   "name" : "sleep"
}
```

Figure 4.5. Example of instantiation of DFA.

In the current implementation, in the same way as the audio contents, the DFA are stored in a remote server (using mongoDB[56]) and the mobile application interacts with it through a web RESTful API. For better scalability, no information about the state of the client app is stored in the server; thus the app will log into the system, tell the server about its context, receive the information about the music that must be played next, and close the connection. Then, if necessary, the app will download the specific fragments of audio by opening a new connection, downloading the content and closing the connection.

In summary, a mobile application is responsible for the direct interaction with the user, counting on a simple user interface and being in charge of playing the music. The app also contains the logic to handle the communication with the server, to download the music and manage the present state of the corresponding DFA, which will be needed to get an update from the server, with the next step to take. And the server act as the content provider and decision maker.

[56] https://www.mongodb.org/

4.4 Applications

In this section we will describe different instances of the adaptive music system, making use of the thousands of themes and hundreds of hypersongs that are accessible in the web repository. Most of these uses consist of applications that are real time responsive to the user or the environment and follow the mechanism based on control by DFA that has been described before. Some others take advantage of the great repository of music accessible from the website, with a more simplified way of working.

4.4.1 Easement of sleep initiation

eMTSD

The name is an acronym for Empathetic Music Therapy for Sleep Disorder and it is aimed to test the adaptive model in the self-management of some types of sleep related disorders like insomnia, somniphobia, sleep apnea, RLS and KLS. These diseases affect more than 20% of the population in developed countries (Stranges, Tigbe, Gómez-Olivé, Thorogood, & Kandala, 2012) and have been linked to the increase of depression and anxiety. The connection of poor sleep quality with hypertension, diabetes and obesity, has also been reported in the literature (Peppard, Young, Palta, & Skatrud, 2000) (Ioja & Rennert, 2012) (Barceló, et al., 2005). For this reason, and since music can help to relax (Lai & Good, 2005) (Huang, Good, & Zauszniewski, 2010), we have designed eMTSD for mobile devices to implement a 14 days trial,[57] to study the benefits of empathetic music in sleep initiation and maintenance.

The app works by placing the smartphone over the mattress, which makes possible to identify the state of sleepiness in the user, characterizing their movements. This will be done by combining the 3-axis data provided by the accelerometer, and then processing and analyzing it.

To use this resulting information, we have design a simple automaton whose purpose is just to provide music with a level of activation proportional to the

[57] www.geb.uma.es/emtsd

detected state, namely, the more relaxed and nearer to fall asleep the user is, the softer and quieter the music will be.

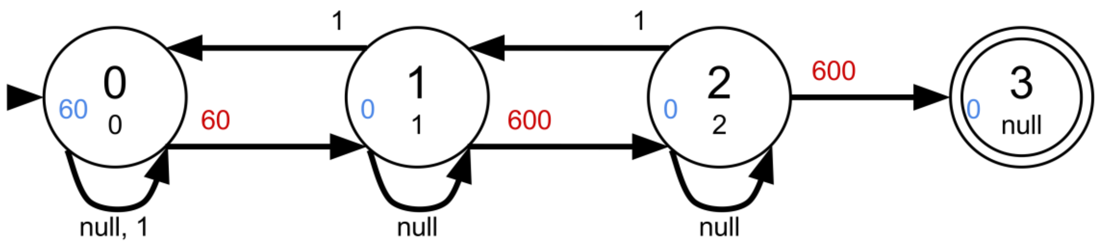


Figure 4.6. Visual representation of the automaton for eMTSD. Each state is represented by a circle, labeled with an ID, the associated output is located below and the insensitivity time to the left, in blue. The arrows represent the transitions, marked with the input or the null-input timeout (red) that triggers them. The initial state is preceded by an arrow with no origin and the final state is illustrated with two concentric circles.

Nevertheless, all versions of the hypersongs used in this application have been designed with the intention of having a low arousal and a positive valence. After being generated, the results were reviewed and approved for its use by the experts that gave the specifications. The main musical features configured to produce these hypersongs comprise: using a small set of instrument playing the roles of melody, five harmonic accompaniments, a percussion of five instruments and a FX role; avoiding high tessituras and using long rhythmic patterns and virtual instruments with a soft timbre; trying the harmony to be simple and familiar, using only major or Dorian modes, with major triad, minor triad and maj9 as the only possible chords; and choosing a very simple structure and textural evolution, favoring the repetition of ideas.

A hypersong for this application is divided in three levels, corresponding to the states of the automaton, excluding the final one, which will not play music (the user is completely asleep). The differences between levels consist of: the highest level includes the whole instrument set (their actual presence is conditioned to the inner textural evolution of the theme) and a higher tempo and volume; in the middle level the percussion roles will not appear, a simpler version of the role bass will be used, the tempo and general volume will be decreased and the set of VSTi will include softer instruments; finally, the third level will only include the role melody and two accompaniments, and the tempo, volume and strength of the selectable virtual instruments will be decreased as well.

- Sample 4.3. Fast evolution of a hypersong for eMTSD.

In the user interface, the app eMTSD includes a simple questionnaire for trial purposes and a binary selector to indicate the type of use: nap or sleep. The user has also access to a small set of buttons: play/pause, forward and settings. The DFA of the nap mode has slights variations in the number of states and the configuration of times.

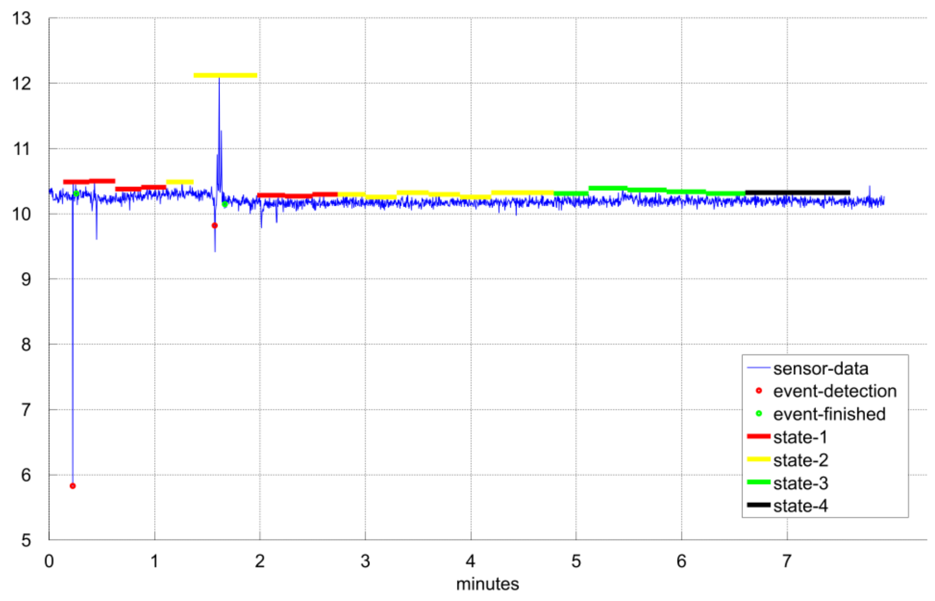


Figure 4.7. Recorded session with eMTSD. Events of motion, detected with the accelerometer, trigger the transition to a higher state of awakeness. When no events are detected, the system automatically moves to lower states, ending in state-4 or asleep.

The application evolved starting from one called @sleep, with a simpler detector of sleepiness, a similar automaton to the one described and a quite loose specification of hypersongs; then the app was embedded as an scenario in a bigger app called @life, where the analysis of the sleepiness was improved and new music was produced, although still with the same specifications. Finally the app was separated to this independent one, called eMTSD, to use in the experiments, with improved specifications of the music. Lately this app was reviewed to promote sleep in babies, giving place to the app named "duermeteya". The software to detect the activity, and the automaton to control the compositional path were similar to that used in the original eMTSD; as well as the specifications for the music, but with simpler structure and harmony, and modifying the chosen instruments, using sounds more recognizable by kids, much alike the ones that appear in regular lullabies.

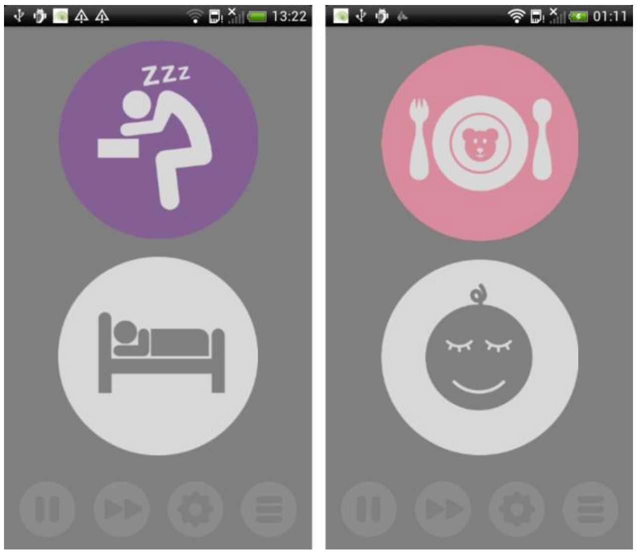


Figure 4.8. User interface of eMTSD and duermeteya.

4.4.2 Improvement of concentration and coordination

PsMelody

Its purpose is to promote performance at work and, at the same time, trying to decrease the level of stress by providing adaptive relaxing music. It is a prototype application implemented in Windows® 7. As input it takes usage data from keyboard (key pressing frequencies) and mouse (movement, clicks and scrolls), to interpret the kind of activity exhibited by the user and the amount of energy that is put on it.

This prototype does not work with the logic in the remote server, all the data and code remain stored locally. The default behavior is implemented by an automaton that simply plays music whose rhythm is related to the activity detected in the user, with two bounds: anxiety, with the music at a lower rhythm than would correspond, and idleness, with the music being played at a bit higher rhythm compared to what is detected. This application comes with the tool JFLAP integrated on its interface, so personalizing and testing new behaviors is very straightforward.

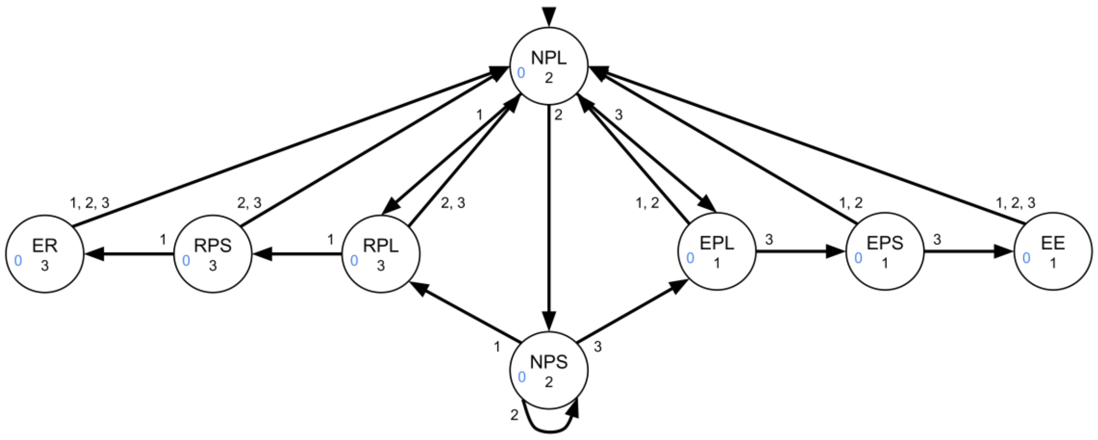
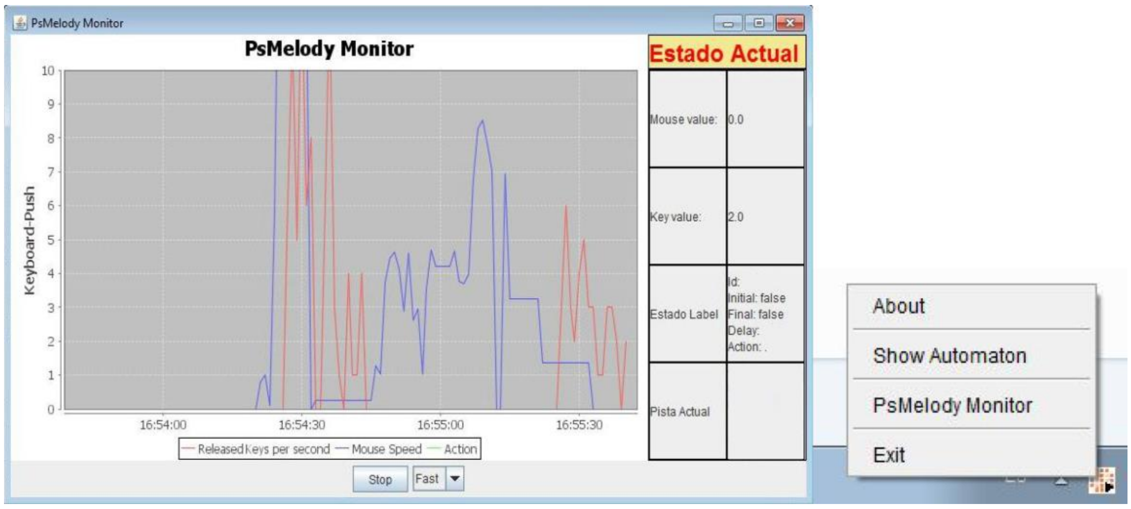


Figure 4.9. Visual representation of the automaton for PsMelody. Each state is represented by a circle, labeled with an ID, the associated output is located below and the insensitivity time to the left, in blue. The arrows represent the transitions, marked with the input that triggers them. The initial state is indicated by an arrow with no origin. The ID are acronyms for: Emergency Relaxed (meaning too relaxed), Relaxed Period Short, Relaxed Period Long, Normal Period Short, Normal Period Long, Excited Period Long, Excited Period Short and Emergency Excited.

In Figure 4.9 is shown the default behavior of PsMelody. The activity detected by analyzing the use of the keyboard and mouse is simplified in three levels (1, 2 and 3) and the hypersong is formed by three versions (also labeled as 1, 2 and 3).



Estado Actual

Mouse value:	0.0
Key value:	2.0
Estado Label	Id: Initial: false Final: false Delay: Action: .
Pista Actual	

Figure 4.10. User interface of PsMelody.

The user interface has been simplified in the form of a tray icon on the taskbar, which can lead to four actions: the “About” option, the window to manage the automaton through JFLAP, the PsMelody Monitor that shows information and graphic representation about the data that is being retrieved and analyzed, and "Exit".

The music used in this application puts some emphasis on the rhythm, to suggest a proper tempo to the user, but trying to prevent from being annoying, since the music shall be played in background while the user is focused on other activities. In general, the music designed is relaxing and comprising a small set of instruments. To handle the different preferences and to prevent from boredom, different musical styles have been considered (pop, rock, jazz, blues, soul, classic, ambient, indie styles…). The tempo will change depending on the current piece of music and the decisions made by the state-machine (globally from 40 to 130 bpm). The harmony progressions and used chords, as well as the rhythmic patterns and global structure, will be simple, since it must not attract much user's attention.

@car

The target of @car (Gutiérrez-Gutiérrez, 2013) is to play music adapted to the driving and the traffic conditions, with the intention of favoring concentration in any circumstance and trying to reduce fatigue, especially under irregular conditions, like extreme weather or saturated traffic.

The characterization method will be based on different information, extracted from some sensors that are present in most smartphones: GPS, accelerometer, photometer and magnetometer. From the raw data, the characterization system will infer information that can be classified in four groups: (1) information about the current road or street, (2) present state and events in the thoroughfare, (3) state and events from the driver and (4) weather conditions. All these states and events will be scored with a positive or negative sign and different magnitudes, depending on how it affects to the activity of driving. The automaton to control the behavior takes these punctuations as inputs, moving through four different established states, each one with an associated level of music.

When starting to drive, the mobile device must be secured in a fixed position in the car, in any desired orientation. Thus, the application is able to initialize, and eventually update, the position of the axes of movement. In the figure below, we show a session recorded while driving first through a residential area, then a highway, a main avenue of the city, and some more narrow streets at the end of the route.

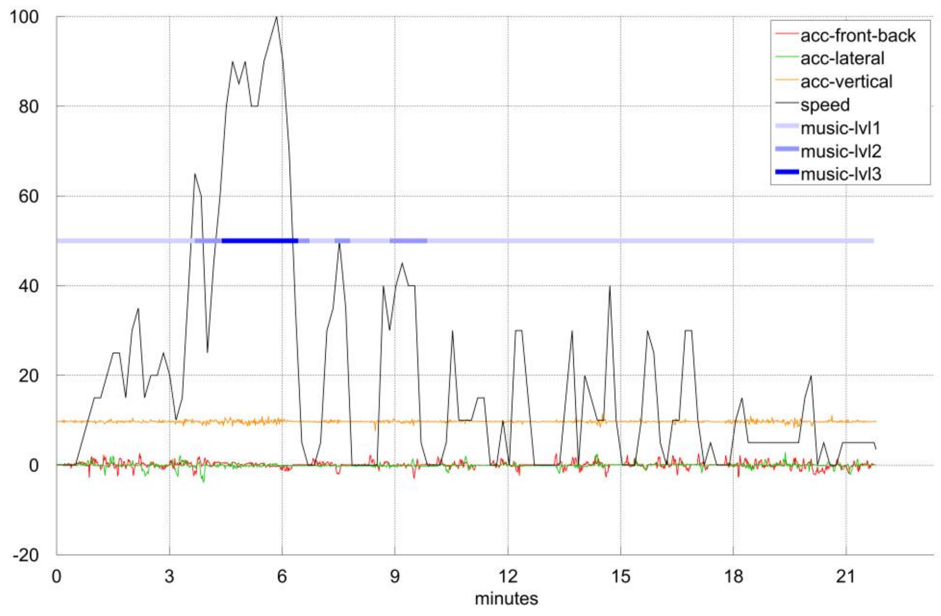


Figure 4.11. Recorded session with @car. The 3-axis accelerometer data are shown in orange, red and green (in m/s^2); the speed measured through the GPS (km/h) is shown in black; and the levels of music played by the system are represented in different tones of blue.

Two relevant aspects about the specified music are: the chosen virtual instruments are softer in the more relaxed states and more percussive in the more activating ones; and volume, rhythm, instrumental density and complexity of the compositions are proportional to the detected state of activation. Different musical genres have been used to attend to the different preferences, but primarily to provide a varied playlist, which would help to avoid monotony during the activity of driving.

@car led us to implement much of the functionality that has been the base for the rest of apps, including the classes to work with the sensors, the overall architecture and the adaptive player, this later being capable of acquiring the musical fragments from different sources (internal resources, expansion files, remote servers…) and following different behaviors (using streaming, caching files, using WIFI connections or using cellular networks…).

Once the app is installed and the extra files have been downloaded (if needed), the interface that shows up is intentionally simple, only with a big play/pause button on the center. The app will provide with the adaptive flux of music, with no need of additional intervention, being more comfortable and safer for the driver.

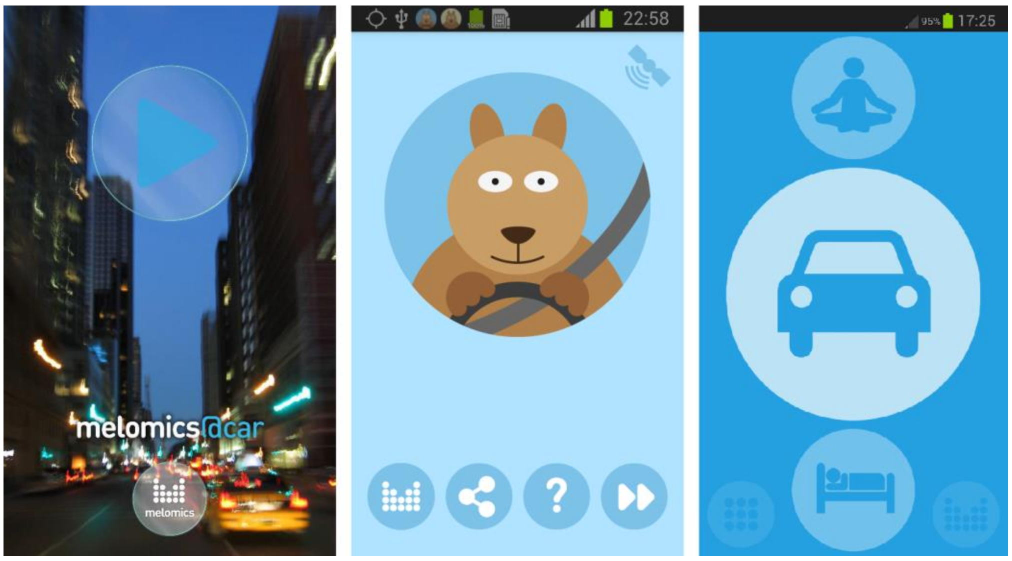


Figure 4.12. User interface of three versions of @car. The original, the most recent *commuting* and the inclusion as a scenario of @life.

@car has been modified in different occasions to produce apps with slightly different purposes. One of the simplest versions, the scenario for the @life application, only uses data coming from the GPS to feed the automata, so the level of music that is played is only based on the speed of the vehicle. There is another version of this app with a simplification in the hypersongs, using only three levels of music. These simplifications and generalizations make this app suitable to use when traveling by other means, such as by bus or train.

@memory

This is an app that makes use of music in order to exercise the associative memory. Counting on music from Melomics and a method for generating simple forms, the app proposes a game based on matching melodies to figures, through a sequence of different stages with increasing difficulty. @memory has an intentionally very simple user interface, with no setting, menu or button of any kind, just the series of questions divided each one in two phases: (a) memorizing associations between melodies and figures, and (b) trying to identify the image corresponding to the song that is being played in this second phase.

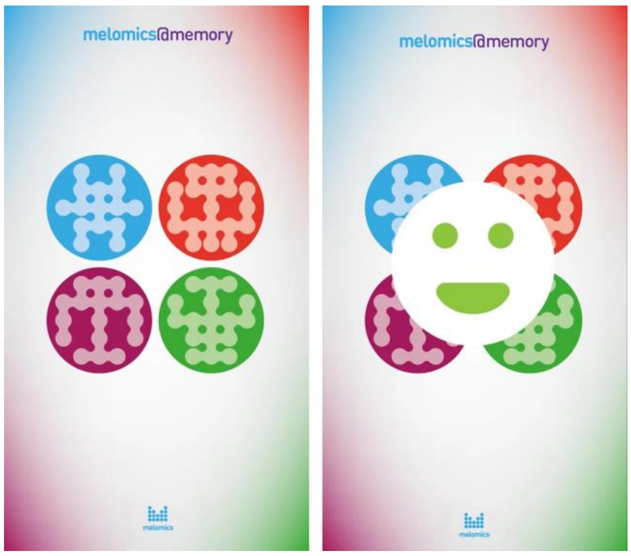


Figure 4.13. User interface of @memory.

The music used in this app was a bundle of short pieces (of few seconds), configured with a relatively high BPM and predictable rhythms, simple and familiar harmonies, a medium-high pitch and a small set of very recognizable instruments, playing a very clear melody with a simple harmonic or rhythmic accompaniment.

Some application and experiments oriented to help in memory related diseases were proposed taking this idea as basis, combining the method with the systems based on adaptive music. Some specific target might be managing emotional states of anxiety and agitation, reactivating decayed cognitive processes, stimulating the maintenance of decayed physical processes, and improving concentration, attention and recognition of people and objects. The diseases of Alzheimer or Multiple Sclerosis are two examples that can be tackled, using this therapeutic approach.

@rhythm

This is a simple app designed to improve concentration and coordination. It presents short pieces of music to the users, so they have to identify the leading beat, indicating it by tapping in a specific point on the screen surface. The logic is very similar to the previous app; it has a simple interface and it gives a series of stages with an increasing difficulty. On each stage, the user must start tapping on the screen as soon as the main beat has been identified and then the user is given a score based on the precision while following the rhythm.

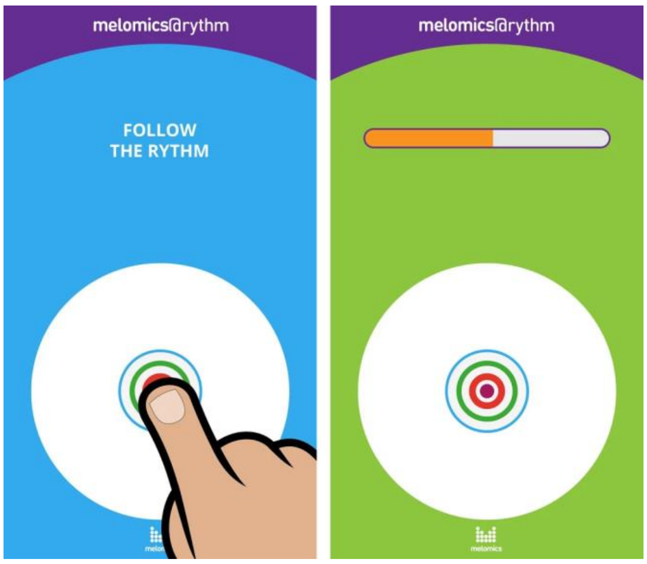


Figure 4.14. User interface of @rhythm.

They were used a bundle of short and simple musical pieces, with a duration from a few seconds to near a minute, depending on the level of difficulty. The global tempo also may vary between 40 to 120 bpm and the ensemble may be formed by 2, 3 or 4 recognizable instruments belonging to the string, wind and percussion families. The tempo will not change within a musical piece, the compositional structure and harmony remain simple and they were used medium-high tessituras, major modes and simple chords.

4.4.3 Management of pain and anxiety

eMTCP

Because music is a powerful distractor, it can be used to reduce pain perception. In order to learn how the music can help in the self-managing of chronic pain, eMTCP (for Empathetic Music Therapy for Chronic Pain) is a mobile app that was originally designed to perform a 14 days ambulatory clinical trial,[58] without the supervision of specialized personnel, or control in a clinical sense. The hypotheses to be tested were: (1) empathetic music therapy improves the quality of sleep in subjects with chronic pain, and this correlates with reduced pain perception and (2) it reduces the perception of pain when applied interactively. Subjects will report through the app with respect to the subjective level of pain perceived and the music will adjust accordingly in real time. This application could be used anywhere and anytime the user feels pain. eMTCP was promoted by the American Chronic Pain Association in order to test the system on its large community of users, involving thousands of people. Chronic pain affects 100 million people in the US alone and it is associated with a large number of conditions (Institute of Medicine (US), 2011). An estimated 20% of American adults report that pain or physical discomfort disrupts their sleep a few nights a week or more (National Sleep Foundation, 2000), so the goal was to create an effective and inexpensive way to manage pain.

In the last version, eMTCP applied to sleep promotion, behaves in a similar way as eMTSD, including the automaton and the data analysis to feed the system. In the rest of the cases, the app will receive feedback from the patient's subjective perception of pain, using a visual numeric scale (Ritter, González, Laurent, & Lorig, 2006) to indicate the current level of pain.

The state machine defined for eMTCP for its use during the day is a little more sophisticated than the others described: it comprises 25 states and 57 transitions among them; it makes extensive use of the parameters null-input timeout and insensitivity; and it considers a wider range of inputs and outputs, with 11 states of pain perception and 8 levels of music respectively. The global interpretation of its behavior can be described as follows: At the beginning, the user is provided with music on its softer version and it keeps working like that while no grade of pain is indicated. If the user marks a level of pain higher than 0 on the visual analog scale (VAS), the automaton starts operating on its standard way, that is, the level of the output will be increased if a new level of pain, slightly higher than the previous one, is marked. In any other case, over

[58] www.geb.uma.es/emtcp

time, the internal state of the system will be gradually decreasing to the lowest level, and the output music will be softer in the same manner.

The main parameters studied by the musical therapists and set in Melomics to produce compositions for eMTCP comprise: (a) a relative high number of roles participating in a composition, including three types of melodies, different kinds of harmonic accompaniments (like pads and chords), two types of arpeggiators, two classes of bass, different roles of rhythmic accompaniments (like a hit-hat, a snare drum, a bass drum, a bongo, a shaker, a conga or a triangle) and two channels of FX; (b) simple harmony, with Dorian and major modes and major, minor, sus2 and maj9 chords; (c) internal accentuations simple and familiar, as well as the allowed rhythmic patterns; (d) style designed to produce long themes, of 10 minutes on average, but with a simple structure in the lower levels, with a very low number of ideas and phrases and very few alterations when these compositional units are repeated; (e) only three predefined models of textural evolution, (1) starting from the minimum number of instruments playing simultaneously, gradually incorporating roles to the maximum allowed number, and then gradually again to the minimum, (2) starting from the maximum, gradually to the minimum and then back to the maximum, and (3) starting from the minimum number, gradually to the maximum, instantly to the minimum and then gradually to the maximum again.

The hypersongs have eight levels of intensity, with lower volumes and tempos for lower levels and vice versa. The textural density will vary from only using one melody and one accompaniment in the softer level, two melodies and some accompaniments in the medium levels, and using most of the accompaniments in the higher levels. Finally, soft VSTi will be used in the synthesis.

- Sample 4.4. Evolution of a hypersong for eMTCP through all its levels.

eMTCP includes a login system, originally designed to manage the ambulatory trial. After the user is sent an email with an identifying code, this and some personal and basic medical data shall be supplied to the app. Then it will start working, presenting a simple interface with big buttons, similar to the apps that have been commented before. eMTCP will collect and send to the server usage and operational data to be analyzed.

In 2013 this application was used as the basis for the system Melomics Music Medicine (M^3), to perform a clinical trial in Hospital Materno-Infantil, Málaga, in the pediatric allergy unit (Requena, et al., 2014). This experiment was aimed to reduce pain perception during the pediatric skin prick test (SPT), ending with a high success compared to the control group, as summarized in Table 4.1. It is planned an analogous experiment at the same hospital in its pediatric onco-hematology unit, pursuing the same goal, during the lumbar puncture procedure.

Pain (VAS)	Experimental	Control	p
No pain (0-2)	22 (71.0%)	7 (18.9%)	<0.001
Moderate acute (3-10)	9 (29.0%)	30 (81.1%)	
Mean/SD	2.1/1.1	4.9/2.7	<0.001

Table 4.1. Pain reported by the two groups during the skin prick test.

After the clinical trial period, the app was released under the open access terms.[59] Later, a simplified version, named *chronic pain*, was addressed to a more general public. The login requirements and the survey for introducing personal and medical data were suppressed and social interaction mechanisms were enhanced. The automaton was reduced, operating with three stages of pain this time and with hypersongs formed by five levels. The application was released during the event SDF2015 celebrated in Seoul, South Korea, in May 2015.

[59] https://en.wikipedia.org/wiki/Open_access

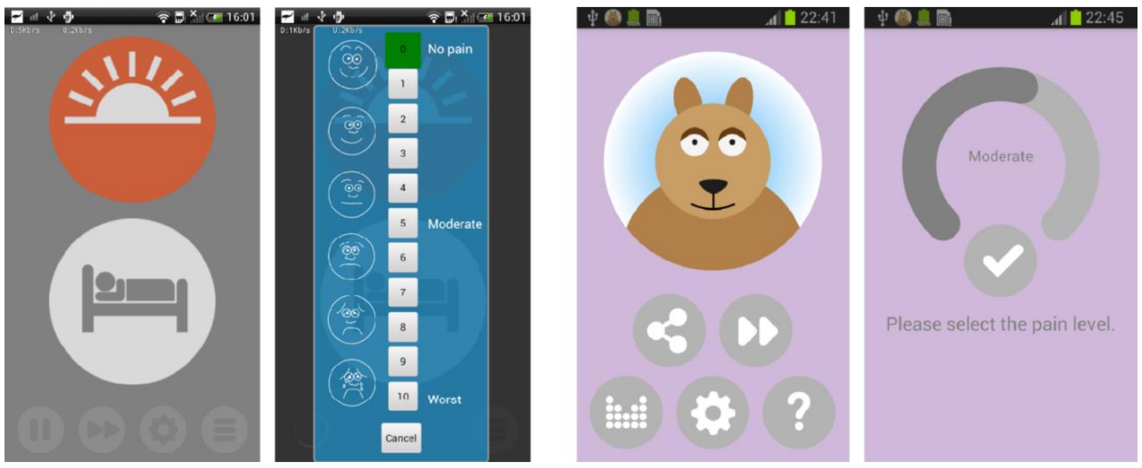


Figure 4.15. User interface of the two versions of eMTCP. The original and the simplified *chronic pain*.

STM

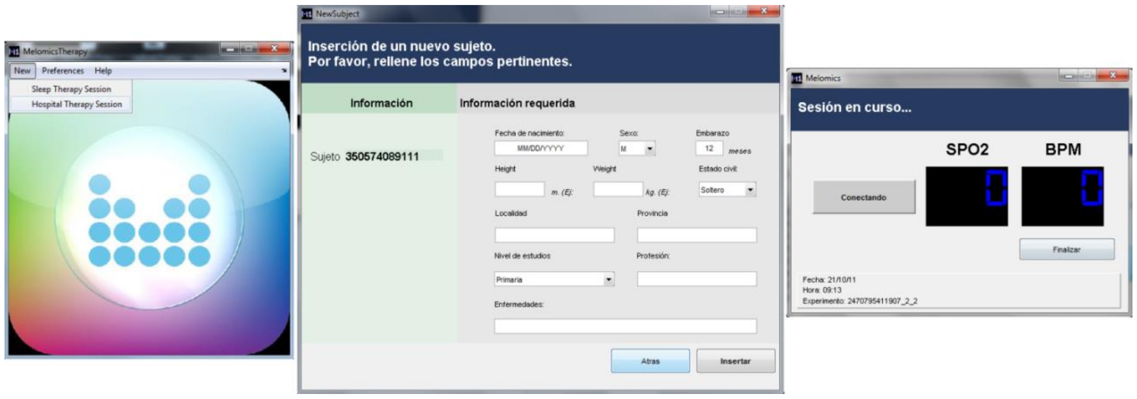
The very first use thought for the adaptive music system consisted of a therapeutic application for hospitals and clinics. This primary system could manage different types of biosensors installed in a PC (a netbook in the trials) and analyze the corresponding biosignals to characterize the state of the patient. The system running in the PC was also in charge of managing the automaton and the adaptive playlist.

In 2010 some members of the research group GEB (Grupo de Estudios en Biomimética) from the University of Málaga were awarded for a project called "MELOMICS, Dispositivos de biofeedback para terapia musical" in the SPIN-OFF contest for start-ups, promoted by that university. The aim of the project was to explore and test this kind of therapies in end-user environments. In late 2010 the GEB was also granted for a research project named "MELOMICS Optimización de la respuesta terapéutica a la modulación de estímulos auditivos"[60] with the purpose of developing and testing the system in hospitals. A summary of this project is given by (Vico, Sánchez-Quintana, & Albarracín, 2011), published in the VIII European Music Therapy Congress in Cádiz, Spain. The main designs and evolution of this therapeutic-oriented adaptive application will be described next.

[60] www.geb.uma.es/research/melomics.html

The system was called STM (Spanish for Sistema de Terapia Musical), later transcribed into English as EMT (Empathetic Music Therapy). It was built to carry out experiments in different units at the Hospital viamed Montecanal, which was a partner in the research project. The starting trial was performed in the neonatal unit and consisted of providing with adaptive music to the mothers of newborns during the breastfeeding. The heart rate of the moms was taken as the input parameter to interpret their state of anxiety; the aim was to favor a low level of stress, which was hypothesized to lead to a low level of stress in the newborns, increasing the breastmilk sucking rate, and thus improving their feeding, especially during the days.

The STM system was prototyped using the MATLAB language. It comprised a small database; a set of classes to acquire the data coming from the heart rate monitor, with the possibility of adding other sensors just by implementing the proper methods; a component in charge of processing the signal and characterizing the patient's current state; the code implementing the state machine model; an early version of the advanced multimedia player, based on two simple media players handling the hypersongs; and a GUI, with an initial survey to be filled before the first session and a window showing the ongoing therapy, with the current heart rate, the oximetry status and some other information about the experiment.



Inserción de un nuevo sujeto.
Por favor, rellene los campos pertinentes.

Sujeto 350574089111

Fecha de nacimiento: MM/DD/YYYY
Sexo: M
Embarazo 12 meses
Height m. (Ej:
Weight kg. (Ej:
Estado civil Soltero
Localidad
Provincia
Nivel de estudios Primaria
Profesión:
Enfermedades:

Sesión en curso...

Fecha: 21/10/11
Hora: 09:13
Experimento: 2470795411907_2_2

Figure 4.16. User interface of the prototype STM.

The double-blind experiment performed included three groups of patients: (1) the group with headset but no music being played at all; (2) the group supplied with static music, unresponsive to the patient's state (a single level of the hypersongs was used for this group); and (3) the group supplied with the adaptive therapy, characterizing the state in real time and deciding the audio content to be played next. The MATLAB program was compiled and stored in an Asus EeePC netbook, connected to the internet through a MIFI connection. The database and the logs from each session were stored in a folder synchronized via Dropbox™[61], so the research teams could access this data remotely.

To characterize the state of anxiety, a method using personalized thresholding of the heart rate was designed: At the beginning of each session, the basal BPM is established by taking measures for a period of time, then the states are computed by dividing the ranges of BPM according to this value:

$f(t)$ is the heart rate over the time t, in seconds.

The basal heart rate is estimated in a time T_B (usually 2 to 5 minutes, $T_B = 120$ or $T_B = 300$).

$f_b = \frac{\sum_{t=1}^{T_b} f(t)}{b}$, is the basal heart rate.

$d = \sqrt{\frac{\sum_{t=1}^{T_b} (f(t) - f_b)^2}{b-1}}$, is the deviation of the data while measuring the basal

$C_e = 0.4$

$C_r = 0.2$

$T_e = f_b + C_e \cdot d$

$T_r = f_b - C_r \cdot d$

The patient is *anxious* if $f(t) > T_e$

The patient is *normal* if $f(t) \leq T_e \wedge f(t) \geq T_r$

The patient is *relaxed* if $f(t) < T_r$

The coefficients C_e and C_r were adjusted in an empirical fashion, with 10 people undergoing approximately 30 tests in total.

[61] https://www.dropbox.com

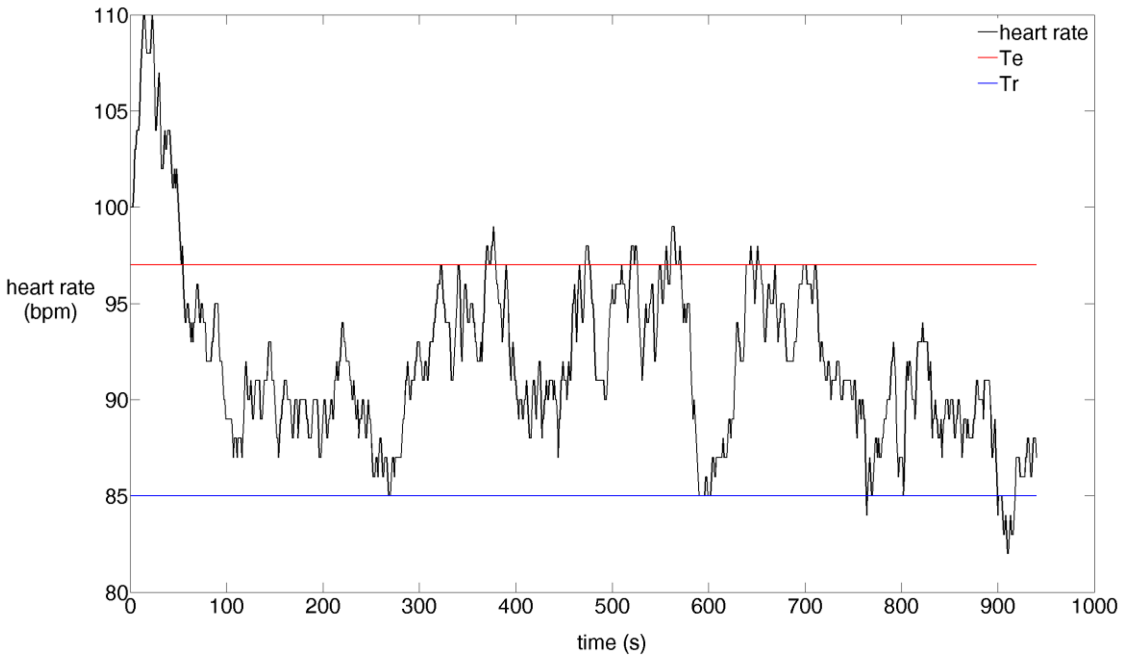


Figure 4.17. Data registered during a session with the STM system.

The automaton to control the music can be divided in M stages (in the experiment $M = 5$), each one representing a different attempt to catch the attention from the user, with the goal of achieving a level of complete relaxation ($f(t) < T_r$). The implemented behavior on each stage would be: If a high level of anxiety is detected, a measure based on acoustics actions will be taken to attract the user's attention, trying to prevent them from focusing on disquieting thoughts. This type of state (E-state) will be maintained for a very short period of time, after which the state machine will move to an N-state. N-states are usually associated with normal input (the present heart rate is near the basal). If the detected state keeps being normal, the automaton will remain in an N-state, but if the system detects a more relaxed state in the user, the automaton will move to an R-state, with an even softer version of the hypersong. As said, the state machine has several stages, each one working in a similar way (same substructure of states), but with different output in the E-states. In our implementation the system carries out the next actions: Stage-1, an abrupt pause of 10 seconds is inserted in the flux of music; Stage-2, the fourth level of the hypersong (not designed to be relaxing) is played; Stage-3, the third level of the hypersong (compositionally relaxing, but with high dynamics) is played; Stage-4, an independent short melody is introduced into the flux of music; and

Stage-5, the hypersong is changed. The automaton will move along the stages every time it passes through an E-state.

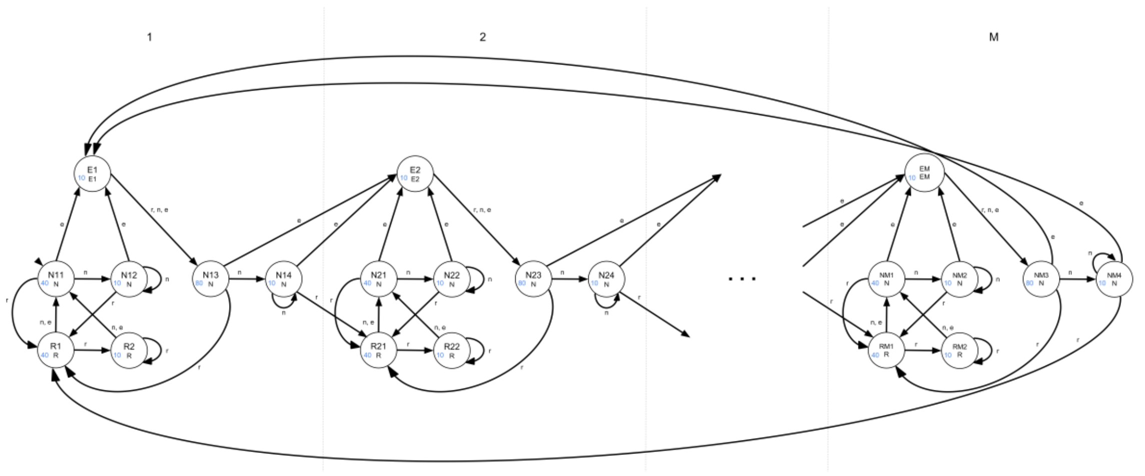


Figure 4.18. Visual representation of a template of the automaton for STM. Each state is represented by a circle, labeled with an ID, the associated output is located below and the insensitivity time to the left, in blue. The arrows represent the transitions, marked with the input that triggers them. The initial state is indicated by an arrow with no origin. Each of the M stages is composed by 2 states of type R (Relaxed), 4 states of type N (Normal) and 1 state of type E (Excited).

The music for STM has been designed, in general, with positive valence and low arousal, according to the directions given by the music therapists. The hypersongs are formed by four levels: level 1, incorporating the softer VSTi and with a very low tempo, volume and textural density; level 2, corresponding to the neutral relaxing layer, with a slightly higher tempo and volume, and a different instrumentation, with a little more sharped attacks and decays of notes; level 3, equal to the second one, but with a higher volume; and level 4, compositionally not relaxing. The participant roles can include two types of melodies, a pad, a role of chords, an arpeggiator and a bass. Major modes are commonly used, but Dorian modes may also appear. The allowed chords comprise major triad, minor triad, sus2 and maj9 and the pitch steps for melodies are kept simples. Low tessituras are favored over high pitches, the different modes of internal accentuation and the rhythmic patterns are quite basic, as well as the textural evolution and the compositional structure, trying to favor repetitiveness.

The STM system for hospitals evolved, acquiring the capability of obtaining data from other sensors and implementing new different state machines. Many other experiments were proposed and performed over patients undergoing unpleasant or painful procedures, for example in the hemodialysis unit of Hospital Costa del Sol in Marbella. STM was also used to propose therapies for treating sleep diseases in some clinics and research facilities.

4.4.4 Sport

@jog

Is an adaptive mobile application that performs pedometry to detect the runner's current leading pace and plays music in real time, according to this information. The interface has been kept simple, only showing the time spent during the current session and a big pause/play button to stop/resume the playback at will. The setting is also rather straightforward; it has a timer to establish the session time, if so desired, and three different behaviors with increasing difficulty. The easy mode consists of the application just playing music to fit the user's rhythm during the session. The regular mode is a variation, where the application computes the average pace and, if the current pace falls under a certain percentage of the average, the played version of the hypersongs will be slightly more intense than will correspond to the state detected, in order to attract the user's rhythm at least the average of the session, thus promoting a higher performance during the exercise. The hard mode includes an additional behavior: at certain moments the system will increase the intensity of the music to encourage a higher performance. This boost state will last for a short period of time, after which the automaton will return to the normal state (with a more intense music if the boost took effect). A visual representation of the state machine for the hard mode is shown below. The regular mode will be identical, just removing the states B, N2 and the corresponding transitions; and the easy mode will consist of only the state N1.

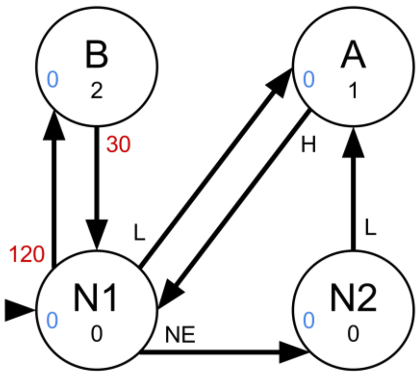


Figure 4.19. Visual representation of the automaton for @jog (*hard* session). Each state is represented by a circle, labeled with an ID, the associated output is located below and the insensitivity time to the left, in blue. The arrows represent the transitions, marked with the input or the null-input timeout that triggers them (red). The initial state is indicated by an arrow with no origin. Input NE is associated with an internal counter of times to encourage a higher pace to the user. Inputs L and H are associated to the present pace with respect to the average pace during the session.

Detecting the leading pace is made by an algorithm that takes as input the data provided by the accelerometer. The device must be placed attached to the body, for example in a tight pocket or with a bracelet for smartphones. The 3-axis signal will be filtered, combined and analyzed, making use of a Short-time Fourier transform and adjusting several parameters, in order to detect the right current rhythm according to the type of movement made by a runner and to achieve the proper balance between precision and battery consumption.

The hypersongs in @jog are formed by the necessary levels to fit the possible rhythms that a runner may follow. The current setting includes 25 levels, where the main changing parameter is the bpm, starting from 80 (corresponding to a slow walk) to 200 (an extremely fast sprint). The appropriate step between bpm, after a large series of tests, was established in 5. The different levels of the hypersongs may also vary in their musical texture. For example, the set of eligible virtual instruments depends on the considered range of bpm, and the participating roles and the prevalence of each instrument may vary too, following similar criteria.

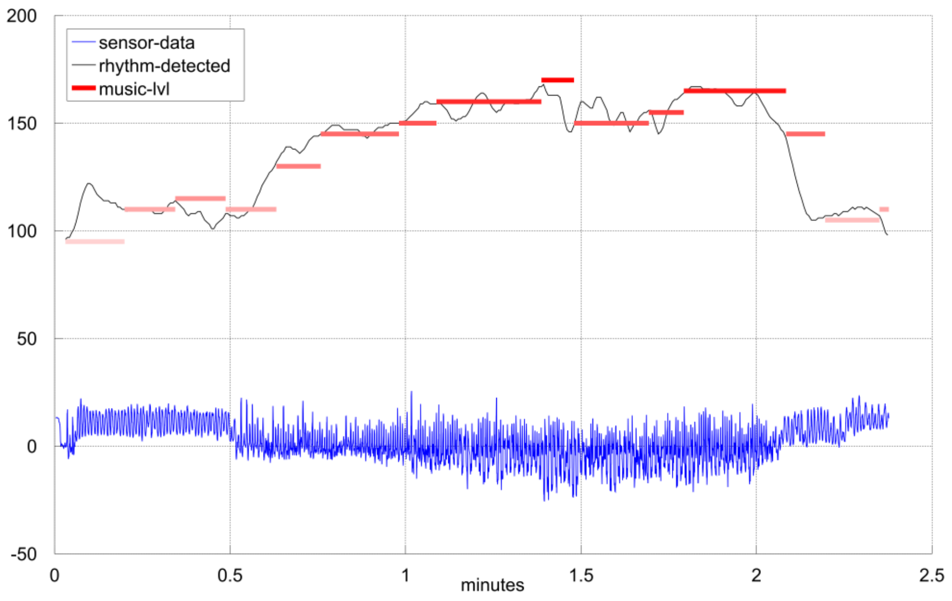


Figure 4.20. Recorded session with @jog, on easy mode. The accelerometer combined data is shown in blue; the real-time detected leading rhythm (beats per minute) in black; and the level of the music played by the system is presented with different tones of red.

The style of hypersongs designed for the @jog have the next features: (a) a set of roles of medium size, including two melodies (used depending on the range of bpm), some harmonic accompaniments (pad, chords and arpeggiator), a bass (rhythmically coordinated with the percussion), a percussion set, and a track for FX; (b) the style uses the major, Mixolydian and Dorian modes, with major and minor triads and sus2, sus4, maj7 and maj9 chords, and different kind of harmonic progressions; (c) the possible compositional structures, as well as the possible textural variations, are very restricted; (d) there is a small number of alterations between compositional units, and pitch shift, tune modulation and modal progression are the only valid operators; (e) one type of musical fill is used in this style, with a duration of 1 or 2 measures, and appearing at certain moments during the composition, to produce a break in the normal evolution, thus creating certain expectation.

Different versions of @jog, with some little variations on its functionality have been released. One of them was an adaptation to be included as a scenario in the app @life. The major difference in that case was the use of the variable speed to estimate the rhythm, using the GPS instead of the accelerometer. This was done to simplify the logic, user permissions and to reduce battery consumption. The last version of this app followed the same behavior as described before, but working with only one level of difficulty, the easy one, and having a button to skip the current hypersong if desired. Some sharing options to the main social platforms were included too.

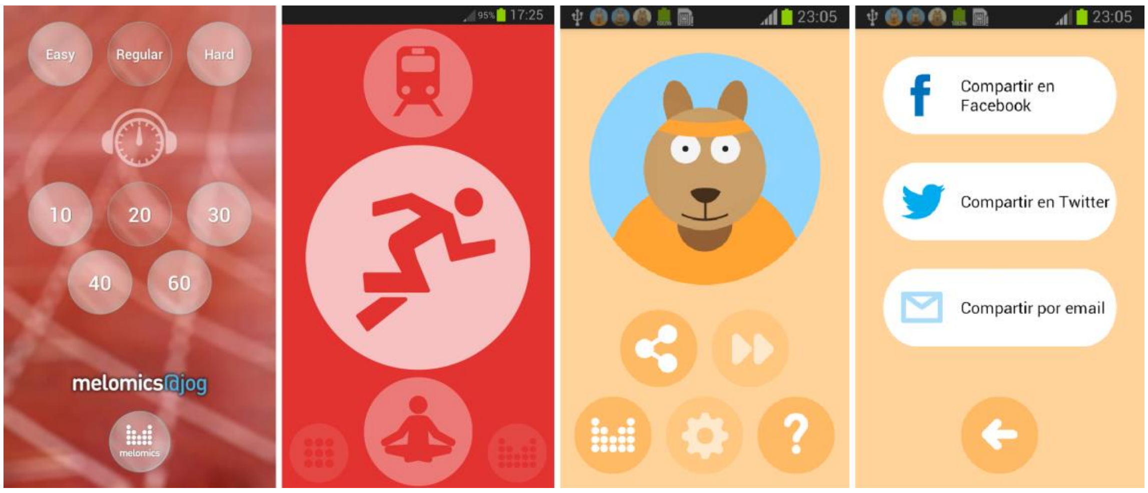


Figure 4.21. User interface of the three version of @jog. The original, its inclusion as a scenario of @life and two captures with the most recent version *sports*.

Other applications have been implemented, taking the main idea from this application, which is detecting the user's pace while doing certain activity and then providing a sort of guide through music, to perform it better or easier; @bike, described next, is an example. And several other applications and research experiments have been proposed for helping in the treatment of different pathologies: hypertension, Parkinson, disabling stroke, Alzheimer, multiple sclerosis, asthma, depression, neuromuscular damages or helping in recovery from surgery involving the locomotor system.

@bike

This application was developed as a variation of @jog; they share the same way of functioning and they have a similar aim: helping to improve the performance during the exercise. The user interface is practically identical to of its precedent app; and the state machines and the method for detecting the rhythm remain the same. The mobile device must be placed in one of the legs and some parameters had to be adjusted, because of the different nature of this cyclic movement. We use the same style of music for this app, because the intention is to induce the same kind of behavior, motivating and encouraging the user to maintain or increase its rhythm. Nevertheless, the number of levels of the hypersongs was extended, in order to satisfy the higher range of rhythms that can be held while cycling.

This app was not included as a scenario in @life, since that app trusted all its functionality on the GPS and in this case there is no correspondence at all between the speed and the detected rhythm, due to the use of the different gears. On the other hand, @bike suffered some revisions too; it last version is included in *sport*, where it is packed along with @jog in a single app with these two scenarios. As in many previous cases, @bike served as inspiration for some other new applications, for example a system to perform classes of coordinated indoor cycling.

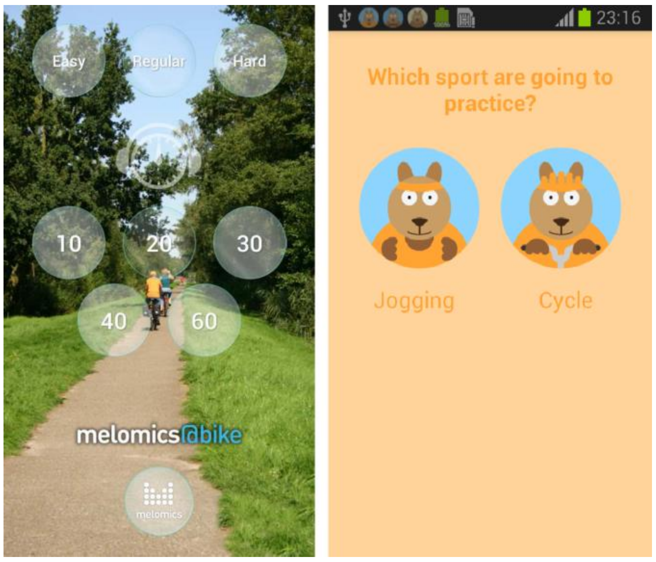


Figure 4.22. User interface of @bike. The original version and the menu to choose between the scenarios *Jogging* and *Cycle* in the app *sport*.

4.4.5 Other uses

The applications presented before are some of the implemented prototypes, although there have been some others, like the apps @trip, *school* or @life, including many different scenarios (*Waking-up*, *Starting-out*, *Walking*, *Break*, *Relaxing*...).

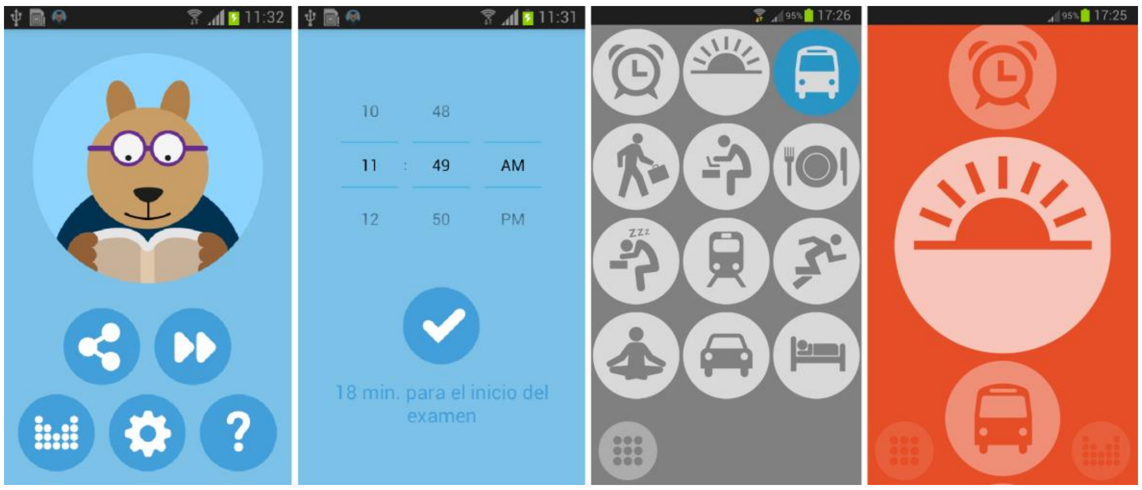


Figure 4.23. Screen captures of different apps. Main screen of the app *school* and its setting menu, the menu of scenarios in @life and the scenario *Starting-out* chosen in @life.

During the execution of MELOMICS research project, the work has been focused on the feedback-based applications, especially the therapeutic ones. Nevertheless, there are several other uses of adaptive music that have been considered, for example its use in video games, where adapted music can help to create a more immersive experience, taking into account thematic, scenarios or real time events. Another related example is its use for creating movie soundtracks. The method would include an automatic or semi-automatic pre-analysis of the frames, in order to establish how should be the evolution of music along the movie. One last proposal is called *IDMelody*, a system consisting of producing short pieces of music that can be owned by people or brands, for example. These small compositions can be indeed hypersongs, comprising different versions. For example people might own an IDMelody whose different levels are used to express their current mood, through the social applications.

Chapter 5

Conclusions

5.1 Summary and contributions

In this dissertation we have presented the compositional system named Melomics. In particular, we have described the design of the two versions of the compositional algorithm, based on bioinspired methods. We have detailed the main mechanisms that are involved during the execution and the way the information is stored along the process. We have shown how music of different styles can be obtained and we have evaluated these results, from several approaches, in relation to the traditional music made by human composers. Finally, we have exposed a way to take advantage of this actual music being produced by a computer, consisting of a completely new way of distributing and playing music.

The main purpose of Melomics compositional system is to generate novel and valuable music compositions, trying to model a creative process, which includes handling knowledge about the field of work (applying rules, expectations...), trying new ideas and approaches, generating effective results and checking these products. We have developed a method based on evolutionary search, supported by two models of implicit genetic encoding, based on formal grammars. The system for atonal music uses a deterministic L-system implementation and it includes a process of development consisting of an iterative phase of rewriting and stabilization, and a final stage where the resulting string of genetic symbols is interpreted into musical elements. The method to create tonal music is implemented with a context-free grammar and it follows a similar developmental process. Besides of producing genuine pieces of music, because of the way they are represented and treated, they are scalable, meaning that any composition can be reused as part of another piece with a more complex structure; the syntax is highly expressive, it has allowed to symbolize any required piece of music with an equivalent representation in the traditional formal music notation; it is flexible, any musical sequence can be written of infinite forms; it is compact, this is, in spite of including all of the compositional and performing information, it consumes between a half and a third less storage than the resultant MIDI file, between a third and a quarter of the MusicXML and definitely less than any audio wave based format; and it is robust, meaning that if a genome is altered in any way, not only it still will produce a valid piece of music, but it also will share many elements in common with the original, it will be a particular mutation of it.

Melomics has produced what can be considered as the first piece of professional contemporary classical music, composed by a computer in its own style, Iamus; it has also produced the first full-scale work entirely composed by a computer and using conventional music notation, Hello World!; a large repository with music of many styles, accessible in the usual symbolic and audio formats, under a CC0 license; and two albums, Iamus album, of contemporary music, and 0Music, with music of some more popular styles.

We developed a tool that allows specifying a musical style by filling the answers to a proposed questionnaire. It includes global values, structural directions and more specific parameters, which Melomics can use to produce genotypes and drive the whole compositional process. The nature of this form makes possible that any person with very little musical background can give directions to the system for creating the desired music. We also tested that when a specific musical target is tackled by different people, they gave similar specifications through the questionnaire, thus producing similar kind of music. The tool was in fact formatted into an online form, whose answers can be directly interpreted by the system, this being the first version of a utility that can serve permanently, to the community of users.

During the development of this whole system, we have acquired some experience, which has been reflected in refining the algorithms and the management of parameters, also settling the foundations for alternate ways of automated search. The acquired knowledge has let us to propose and implement a new representation of symbolic music, which contains compositional information, instructions about the performance and a lot of meta-information, including that generated during the compositional decision making process, resulting in a highly expressive and yet compact music format.

The interactive nature of the Turing Test, still the reference method to evaluate artificial intelligent systems, makes it less suitable to assess algorithmic compositional systems. In this thesis we have presented a more practical test, where the interrogator is replaced by surveys of musical judgements. In particular, the experiment proposed included the analysis from expert musicians and non-musicians, questions about the emotions and mental representations evoked, and a direct query about who composed the pieces. The study reflected that there was no different between the music by Melomics and the equivalent from human composers.

Finally, we have presented the hypersong, a musical structure comprising a musical theme and a bundle of versions of it, all of them fragmented in the time, giving place to a two-dimensional array of music. This concept of hypersong, let us to develop a number of applications based on creating an adaptive flux of music, which is only practical if it can be created by automated means. This technology was also made publicly available in form of an API, fed by a repository of standard music, a bunch of hypersongs and the corresponding meta-information, making possible to the community of developers to build new applications.

5.2 Future work

Due to the multidisciplinary nature of the presented work, there are a multitude of issues that should be tackled in the future, both commercial applications and research lines. Here we present short descriptions of the three main directions that we would want to follow to continue this research.

5.2.1 Genetic algorithm

Along the evolutionary algorithms, the genetic algorithms (GA for short) (Holland, 1975) are one of the most popular and the closest variation to biological evolution. As an alternative to the implemented evolutionary system, we propose to test a more typical GA, where the entire genotypes of the fittest individuals are stored, and then mutated and/or combined to produce the next generation of solutions. Developing this new GA will consist basically of (a) establishing the genetic operators; we have explored this matter in the experiment described in the section, Genetic engineering: Nokia tune and, Genetic engineering: SBS Logo on both the atonal and tonal system; and (b) building a more comprehensive kind of fitness function, by grouping many of the rules that have been incorporated along the different stages of the compositional process.

Our proposal would make use of the current developmental process from a genome written in an implicit manner, which as commented before, is much more suitable to tackle complex problems. This would be precisely the purpose of implementing the alternative GA, to open the range of musical parameters to produce the first population of compositions and then to refine the outcomes over generations with a biased evaluation function, which is intended to give place to more unexpected and creative solutions.

GA have been used before in algorithmic composition, however, they mostly used a collection of some simplistic features, which led to modest results, or the algorithms were focused on producing certain musical elements, not the composition itself, for example (Johnson, Tauritz, & Wilkerson, 2004), (Garay Acevedo, 2004) or (Lozano, Medaglia, & Velasco, 2009). An alternative way to build the fitness function would be to evaluate the distances to a certain targeted compositions or high quality samples, see for example (Gartland-Jones, 2002), (Alfonseca, Cebrián Ramos, & Ortega, 2005) or (Spector & Alpern, 1994). This would be useful to incorporate to the system the ability of composing by imitating styles. The assessment can be run with MIR techniques, for example, similarly as described in 3.2.1.2 Compositional analysis of the produced musical styles.

One last interesting approach will be using an interactive fitness function, also with some attempts in the past (Biles, 1998) (Tokui & Iba, 2000). Our proposal would be taking advantage of the web platform that was developed, as a crowdsourcing system by extracting information from the users comments, the themes they buy or the different scores they would give through a well-designed tool that should be both attractive to the user and capable of providing us with the useful information.

5.2.2 Reaching new styles and musical genres

In spite of Melomics capability of representing any kind of musical content, in practice it only produces music in the styles that have been specified, which is far from covering the entire known musical spectrum yet. It will be necessary to count on more expert knowledge, to define a proper way to explore the rest of the search space, to achieve this known music and new styles yet to be discovered.

Examples of specific settled musical genres that would require further study with the help of experts are: jazz, country, rap, rhythm and blues, rock, folk or many other kind of popular music. Some of them would only require an extended formal analysis to be into the system, but some others come with a bigger problem, they include vocal tracks. We have little experience on introducing voice to compositions, only a few experiments in choral music (Sample 5.1), where the lyrics consisted either of a sequence of "Aahs" and "Oohs" or random fragments chosen from some available texts (like the bible). A complete work of introducing voice will come with the already resolved issue of building the notes for the melody; the problem of building the lyrics, counting on several ongoing researches (Pudaruth, Amourdon, & Anseline, 2014) (Malmi, Takala, Toivonen, Raiko, & Gionis, 2015); and the major and far from being resolved problem of vocal synthesis, much more complex than the normal speech synthesis.

- Sample 5.1. Experiment in choral music.

5.2.3 Personal music

With “personal music” we refer to a way of providing a flux of music that is real time responsive to any context and fits the user’s preferences, hence being the music unique for everyone. This concept of music could be achieved by extending the methods described in Chapter 4, and we have made some first attempts with the app called @life. To carry out this task, it would be required a further study on the proper music for each possible activity and improving the capability of the system to manage a wider range of musical styles and genres.

Some features that will be necessary to study and develop are: (1) a mechanism allowing to detect the user’s preferences in a frictionless fashion and, at the same time, being able to propose new kind of music; (2) a tool exploiting further the hypersong concept, with new ways of versioning songs and new mechanisms to control the flux of music, which currently we only base on simple state machines; and (3) methods to properly detect the present user’s context; besides disposing of tools to characterize the present state on a given situation, we propose a higher level system to characterize the type of activity in which the user is involved, to be used to decide when not to play music and when to shift to a certain scenario, responsible for controlling the concrete flux of music.

Appendix A

Implementation details

A.1 Genetic representation in the atonal model

In the atonal version of Melomics system, the genome of a composition is implemented with a text file identified with a theme name and the extension ".gen". This text file is organized in two levels, the first line correspond to three simple general parameters separated by blank spaces, read inside MATLAB as of type double, and they are: the initial iteration value for the grammar; the initial duration, which also serves as duration of reference; and the duration step length, which is the amount of change applied to the current duration, when the corresponding genes appear in the interpretation of the resulting string. The second level of this model is formed by the subsequent lines, each of them containing five track parameters and a production rule. The parameters are separated by blank spaces and read as doubles. They consist of (1) the iterativity parameter, indicating the way the production rule will stop being applied (as indicated in the section 2.2.2.2 Model of development); (2) the duration factor, affecting the duration of the notes in the corresponding track; (3) the instrument ID, to fetch the musical instrument to perform that track; (4) the musical scale to be applied during the interpretation into musical notes, codified as an integer; and (5) tessitura, or range of allowed MIDI pitches for the track, expressed as two integers. The production rule is only represented by its right side, since the left side is formed only by one symbol and appearing only once, because of being a deterministic grammar. We code these symbols with the reserved character "#" plus an identification number (#0, #1, #2…), so we place the production rules sorted by this ID in ascent order.

Next we will introduce the different symbols that can appear in the production rules and their respective meanings in the interpretation of the final string. These symbols or genes are coded as a reserved character followed by a numeric ID, because there were not enough single characters in the keyboard to represent all of them.

#	Used to represent structural units or notes (see Section 2.2.2). No combination is reserved, but as standard procedure we use #0 to introduce a simple zygote-rule with some initial values, #1 to represent the channel for musical rest and #2 to represent the global structure of the composition. In the interpretation of the resulting string, this kind of genetic symbols (a sub-string formed by the character "#" plus a numeric ID) will produce a musical note in the associated track, acquiring the current values of the whole set of musical properties.
$ ID	Used to code the music operators (see Section 2.2.2).
1	Increase pitch value one step in the scale.
2	Decrease pitch value one step in the scale.
3	Increase duration one step (according to the global parameter).
4	Decrease duration one step (according to the global parameter).
5	Save current pitch and duration values in the stack PS.
6	Return to the last value of pitch and duration in PS.
7	Save current time position in the stack TS.
8	Return to the last time position saved in TS.
9	Apply the articulation legato to the next compositional element.
10	Apply the articulation legato until indicated or the end.
11	Terminate the application of legato.
12	Apply the articulation portato to the next compositional element.
13	Apply the articulation portato until indicated or the end.
14	Terminate the application of portato.
15	Apply the articulation staccato to the next compositional element.
16	Apply the articulation staccato until indicated or the end.
17	Terminate the application of staccato.
18	Apply the articulation accento to the next compositional element.
19	Apply the articulation accento until indicated or the end.
20	Terminate the application of accento.
21	Apply the articulation tenuto to the next compositional element.
22	Apply the articulation tenuto until indicated or the end.
23	Terminate the application of tenuto.
30	Apply the effect-1 to the next compositional element. The concrete effects will depend on the actual instrument.
31	Apply the effect-1 until indicated or the end.
32	Terminate the application of effect-1.
33	Apply the effect-2 to the next compositional element.

34	Apply the effect-2 until indicated or the end.
35	Terminate the application of effect-2.
36	Apply the effect-3 to the next compositional element.
37	Apply the effect-3 until indicated or the end.
38	Terminate the application of effect-3.
39	Apply the effect-4 to the next compositional element.
40	Apply the effect-4 until indicated or the end.
41	Terminate the application of effect-4.
42	Apply the effect-5 to the next compositional element.
43	Apply the effect-5 until indicated or the end.
44	Terminate the application of effect-5.
45	Apply the effect-6 to the next compositional element.
46	Apply the effect-6 until indicated or the end.
47	Terminate the application of effect-6.
61	Apply modulator of dynamics crescendo to the next compositional element.
62	Apply crescendo until indicated or the end.
63	Terminate the application of crescendo.
64	Apply modulator of dynamics diminuendo to the next compositional element.
65	Apply diminuendo until indicated or the end.
66	Terminate the application of diminuendo.
67	Apply fermata to the next compositional element.
68	Apply fermata to all following notes until indicated or the end.
69	Terminate the application of fermata.
70	Apply the property chord to the next compositional element. Instead of a single note, the corresponding instrument will perform a chord, which will be built depending on instrumental parameters and the current value of the “property argument” (see meaning of “@” below).
71	Apply the property chord to all notes until indicated or the end.
72	Terminate the application of chord.
73	Apply the property grace to the next compositional element. Instead of a single note, the corresponding instrument will perform an additional grace note, which will be built depending on instrumental parameters and the current value of the “property argument”.
74	Apply the property grace to all notes until indicated or the end.
75	Terminate the application of grace.
92	Apply dynamic pianississimo.
93	Apply dynamic pianissimo.
94	Apply dynamic piano.
95	Apply dynamic mezzo-piano.
96	Apply dynamic mezzo-forte.
97	Apply dynamic forte.
98	Apply dynamic fortissimo.
99	Apply dynamic fortississimo.
100	Save value of pitch and duration in the stack GS.
101	Return to the last value of pitch and duration saved in GS.
102	Apply the tempo indicated with the current value of the “property argument”

110	Apply the property arpeggio to the next compositional element. Instead of a single note, the corresponding instrument will perform an arpeggio, which will be built depending on instrumental parameters and the current value of the "property argument".
111	Apply the property arpeggio to all notes until indicated or the end.
112	Terminate the application of arpeggio.
113	Apply mordent to the next compositional element.
114	Apply mordent to all following notes until indicated or the end.
115	Terminate the application of mordent.
116	Apply inverted mordent to the next compositional element.
117	Apply inverted mordent to all following notes until indicated or the end.
118	Terminate the application of inverted mordent.
120	Apply pitch alteration in the next compositional element, to the position and with a value codified with "property argument".
@	The numeric string following this character will be considered as the current value of the "property argument", which will serve as a parameter for some musical properties.

Table A.1. Reserved symbols in the atonal genomic model.

A.2 Genetic representation in the tonal model

In the tonal version of Melomics system, the genome of a composition is implemented with a text file identified with a theme name and the extension ".evg". This text file is organized in different levels by the separator "|" and each level with sections divided using the character ";". The first level and first section comprehends the global parameters: initial dynamic, initial tempo, initial chord, initial mode, base duration, duration step length and musical style, read in MATLAB as doubles. Each of the next sections in the first level corresponds to a musical track with all the needed parameters (MIDI ID, initial pitch and tessitura, range of velocities, role ID and instrument ID). The following levels of this model of genotype correspond to each level of the grammar. Each level will comprise as many production rules (represented only by their right side, as in the atonal model) as different non-terminal symbols there are in all production rules in the previous level (the first grammar level has only one rule); they will be matched in order of appearance.

Next we will introduce the different reserved symbols that can appear in the production rules, accompanied by their respective meanings in the interpretation of the final string.

[	Save in a stack the current value of pitch, harmonic root and duration.
]	Restore from the stack the last value of pitch, harmonic root and duration.
$<$	Save the current time position, value of pitch, harmonic root and duration in a stack.
$>$	Restore from the stack the last saved time positon, value of pitch, harmonic root and duration.
N	Increase 1 the counters pitch and harmonic root.
n	Decrease 1 the counters pitch and harmonic root.
T	Increase 1 the current tone shift.
t	Decrease 1 the current tone shift.
R	Increase current duration one step (according to the global parameter).
r	Decrease current duration one step (according to the global parameter).
$M\ index, notes, shift, behavior$	The next instrument symbol will interpret, instead of the normal computed pitch, the chord resulting from applying the properties of the current harmony and the parameters coming with this operator. $index$ fetching the concrete kind of chord defined in the musical style; $notes$ codifying the actual notes to be played of the selected chord, $shift$ indicating a possible additional-local tonal shift and $behavior$ providing further directions to compute the resulting notes.
$v\ accent$	Apply the indicated $accent$ to the next note or chord.
$W\ origin, target$	If $target$ is 0, apply dynamic indicated by $origin$ until indicated or the end. If $target$ is a positive value, apply progressive dynamic modulation from $origin$ to $target$ until indicated or the end.
$C\ role, position, type$	Apply a rhythmic alteration indicated by $type$, in the track indicated by $role$ and note given by $position$.
$V\ role, position, type$	Transform a note into rest or vice versa in the track indicated by $role$ and note given by $position. type$ provides directions for building the note if the fetched position was a rest.
$p\ role, position, n, t$	Apply pitch alteration to the note fetched by $role$ and $position$ with a variation in the scale indicated by n and a tonal shift indicated by t.
$w\ start, end$	In the rewriting of the next non-terminal, only the symbols in the production rule enclosed by $start$ and end will appear.
$X\ role$	Prevent the emergence of the track indicated by $role$ in the next compositional element (rewriting of the next non-terminal).
$Y\ durTotal, durAnacrusis$	Produces a musical anacrusis, deleting the needed

	previous code in the string, which is computed by using the given arguments.
$\$\, code, args$	Apply a predefined musical operation coded in the parameter $code$ with the arguments given by $args$.

Table A.2. Reserved symbols in the tonal genomic model.

In this model, to improve the reading of genomes, we use non-reserved keyboard characters to code both the non-terminal symbols and the terminal symbols corresponding to the musical tracks. This is because we can dispose of enough keyboard characters, since the additional less common musical properties or effects are coded using the operator "$" and because there is an explicit hierarchy in the grammar with no backtracking, so these non-reserved symbols can be reused from one level to another.

Appendix B

Computers used

B.1 Workstation

For daily work, mainly including bibliographic exploration, software development with MATLAB and Python, and performing tests, we made use of a high-end workstation described in the table below, connected through a high-speed internet connection provided by the university, which facilitates to move the required high amount of musical data between the different computers and clusters and to gain access to software and bibliographic subscriptions. This workstation runs Windows 7 as main operating system and a Debian GNU/Linux inside a virtual environment.

Processor	Intel® Xeon® ES645 2.4GHz, 6 cores, 2 threads per core
Memory	DDR3 Triple channel, 12GB
GPU	NVIDIA® Quadro® 4000, GDDR5 2GB
Hard Drive	Main system: 500GB Work directory: 2x2TB with a RAID 1
Network	Gigabit Ethernet

Table B.1. Workstation specifications.

B.2 Computer clusters

Due to the high computational cost of producing and saving musical compositions in the different formats (tens of thousands of them are available in the web repository and even more are stored as a backup repository and for testing purposes), we designed and counted on two different computer clusters.

Iamus is a cluster formed by 11 nodes with 352 cores, 704GB of main memory and 70TB of storage, which we generally use to run the atonal system. It is connected through the university network and accessible via a SSH interface. It runs a Debian GNU/Linux in all of its nodes and uses TORQUE and MAUI to manage the resources and to schedule the tasks, mostly consisting of MATLAB and Python processes. For an extended description see (Perez-Bravo, 2014).

		Per node	TOTAL
Processor	AMD Opteron™ 6128 2GHz, 8 core	4 (32 cores)	44 (352 cores)
Memory	DDR3	64GB	704GB
Storage	Master node Worker nodes	10TB 6TB	70TB
Network	Gigabit Ethernet Fast Ethernet for IPMI		

Table B.2. Iamus specifications.

The second cluster, named Melomics109,[62] was mainly designed to run the tonal version of Melomics system and it was installed in Centro de Bioinnovación in the PTA (Parque Tecnológico de Andalucía)[63] and assumed as part of the university supercomputing network. It is a cluster with 40 nodes, comprising 960 processors, 3840 GB of main memory and 320 TB of total storage using a RAID 1+0.

[62] https://en.wikipedia.org/wiki/Melomics109
[63] http://www.scbi.uma.es/

		Per node	TOTAL
Processor	AMD Opteron™ 6176 2.3GHz, 12 core	2 (24 cores)	80 (960 cores)
Memory	DDR3 PC3-10600R-9 RDIMM	96GB	3840GB
Storage	2TB 6G SAS 7.2K 3.5in DP MDL HDD	4x2TB	320TB
Network	Gigabit Ethernet, 3 ports		

Table B.3. Melomics109 specifications.

Appendix C

Execution statistics

C.1 Atonal system

The internal musical style "CtClarinet" (for Contemporary Clarinet) is used in this test to show-case the lightest possible run in terms of computational cost, with Melomics atonal system. This configuration gives place to compositions with a duration ranging 160 to 195 seconds and having only one track played by the clarinet. The other style used in the test, representing the most computationally expensive style in the atonal model, is "CtSymphonicOrchestra" (for Contemporary Symphonic Orchestra), which gives place to orchestral pieces with a duration ranging 120 to 320 seconds and participating between 25 to 43 instruments.

In this section we show the times spent by the two computer configurations (workstation and cluster) in producing one composition, using a single core and one thread. Measures for the cluster configuration are taken only from Iamus, since this and Melomics109 count on a very similar architecture. We provide times to reach different targeted output, in each case, considering that we produce only the previously required formats, whose dependencies are listed below:

- GEN is the first form of compositional representation; therefore it does not require any previous representation.

- INT means the internal representation, which can be stored as a MATLAB or a Python variable. It only requires the GEN to be written before.
- XML is the MusicXML file, which can be loaded from any musical editor supporting this standard. It requires the internal representation.
- PDF is the printed score that can be directly read by musicians. It is produced starting from a MusicXML file, but this differ from that provided as the standard MusicXML to be imported in the musical editors, so it depends directly on the INT format.
- MID is the MIDI format of symbolic notation; it depends on the INT representation.
- WAV is the first representation of a composition as audio. It uses the MIDI standard as an intermediate step, but these files are different from the MID format that is provided as an output, so it depends directly on the INT representation.
- MP3 is the compressed audio obtained from the WAV file.

Average, maximum and minimum values are taken after 50 runs.

		GEN	INT	XML	PDF	MID	WAV	MP3
CtClarinet	AVG	114.87	121.40	122.50	137.68	126.76	139.01	147.27
	MAX	134.57	140.92	142.05	157.51	146.30	158.64	166.91
	MIN	93.13	99.59	100.64	117.30	104.77	116.84	125.19
CtSymphonic Orchestra	AVG	256.20	394.45	419.91	534.70	462.42	953.01	1033.67
	MAX	562.72	743.17	775.28	912.86	825.26	1341.09	1633.25
	MIN	17.46	125.03	139.37	221.97	166.03	559.81	597.12

Table C.1. Times spent, running the atonal system in a workstation. Time in seconds registered in the workstation (Intel Xeon E645) to generate one single composition in the styles CtClarinet and CtSymphonicOrchestra.

		GEN	INT	XML	PDF	MID	WAV	MP3
CtClarinet	AVG	124.89	130.16	131.79	151.58	136.35	160.30	167.96
	MAX	144.54	151.16	151.26	170.38	158.69	178.78	186.97
	MIN	99.59	106.57	107.89	127.66	112.56	133.57	143.16
CtSymphonic Orchestra	AVG	282.50	427.53	453.22	576.07	492.21	1011.11	1092.35
	MAX	605.16	781.26	818.05	959.71	870.49	1422.04	1716.48
	MIN	23.11	136.24	150.54	241.47	180.62	598.90	635.71

Table C.2. Times spent, running the atonal system in a computer cluster. Time in seconds registered in the cluster Iamus (AMD Opteron 6128) to generate one single composition in the styles CtClarinet and CtSymphonicOrchestra.

C.2 Tonal system

The internal musical style "Relax" is used in this test to show-case the lightest possible run in terms of computational cost with Melomics tonal system. This configuration gives place to compositions with a very simple structure, only 5 musical roles and a duration ranging 100 to 600 seconds. The other style used in the test, representing the most computationally expensive style in the tonal model, is "Disco02", which gives place to compositions in the Melomics disco style, comprising from 19 to 22 different roles, a duration between 90 to 210 seconds and a very complex compositional architecture.

C.2.1 Generation of one composition

In this section we show the times spent by the two computer configurations (workstation and cluster) in producing one composition, using single core and thread. Measures for the cluster configuration are taken only from Iamus, since this and Melomics109 count on a very similar architecture. We provide times to reach different targeted formats, in each case, considering that we produce only the previously required representations (dependencies listed in Section C1). MID_Synth format makes reference to a MIDI file that is used together with some information in the internal representation, to produce the WAV audio file. Statistics for audio synthesis are not given in this case, since this component is similar to the atonal synthesis subsystem and the new tool based on VST™ is under development. Average, maximum and minimum values are taken after 50 runs.

		GEN	INT	MID	MID_Synth
Relax	AVG	0.40	24.90	31.74	31.74
	MAX	0.81	60.52	69.52	69.52
	MIN	0.23	5.44	9.72	9.72
Disco02	AVG	18.59	387.73	773.01	436.30
	MAX	25.37	1117.00	2210.75	1204.90
	MIN	11.25	77.58	159.24	102.83

Table C.3. Times spent, running the tonal system in a workstation. Time in seconds registered in the workstation (Intel Xeon E645) to generate one single composition in the styles Relax and Disco02.

		GEN	INT	MID	MID_Synth
Relax	AVG	0.40	37.35	47.56	47.56
	MAX	2.26	143.65	158.66	158.66
	MIN	0.23	6.58	13.63	13.63
Disco02	AVG	17.80	387.13	796.92	427.73
	MAX	35.48	1249.22	2525.67	1308.68
	MIN	13.85	111.00	237.88	139.38

Table C.4. Times spent, running the tonal system in a computer cluster. Time in seconds registered in the cluster Iamus (AMD Opteron 6128) to generate one single composition in the styles Relax and Disco02

Statistics for MID and MID_Synth representations in the style "Relax" are equal because in this case the file used in the synthesis is exactly the same.

C.2.2 Massive generation

In this section we show the performance (in compositions per hour) that the two computer configurations (workstation and cluster) can achieve, using parallelism. Measures for the cluster configuration are taken only from Iamus, since this and Melomics109 count on a very similar architecture and thus the differences will be a factor of the size of their hardware. We provide measures taken in separate executions, on each one targeting a different format as the final output, only producing the required previous representation (dependencies listed in Section C1).

	GEN	INT	MID	MID_Synth
Relax	16951.15	593.39	414.38	414.38
Disco02	475	40.29	28.95	29.06

Table C.5. Massive generation in a workstation. Average number of compositions per hour using the workstation (8 of its 12 logical cores) in the styles Relax and Disco02.

	GEN	INT	MID	MID_Synth
Relax	1708060.70	24190.43	17495.62	17495.62
Disco02	43188.28	2102.84	1235.59	1600.72

Table C.6. Massive generation in a computer cluster. Average number of compositions per hour using Iamus (352 cores) in the styles Relax and Disco02.

Statistics for MID and MID_Synth representations in the style "Relax" are the same because in this case the file used in the synthesis is the same.

Appendix D

Spanish summary and conclusions

En esta sección se proporciona un resumen en español de las líneas desarrolladas en los capítulos principales del documento, así como las conclusiones y resultados obtenidos a la finalización de la tesis.

D.1 Resumen

D.1.1 Introducción

La creatividad, o habilidad de producir nuevas y valiosas ideas, comúnmente se asocia al ser humano, pero existen muchos otros ejemplos en la naturaleza donde se puede dar este fenómeno. Inspirado por este hecho, en las ciencias de la computación se han desarrollado numerosos sistemas con propósitos específicos que producen resultados mediante un enfoque creativo.

La música, una forma de arte muy extendida en toda la historia de la humanidad, es el principal campo tratado en esta tesis, intentando aplicar los procedimientos que generan diversidad y creatividad en la naturaleza y en computación.

Proponemos un método de composición basado en búsqueda evolutiva con codificación genética de las soluciones, las cuales son interpretadas durante un complejo proceso de desarrollo que incluye reglas basadas en conocimiento experto en los distintos niveles de su arquitectura. Este sistema bioinspirado muestra su capacidad creativa y versatilidad en numerosos casos de uso reales y que es indistinguible de la música hecha por humanos, tanto desde una perspectiva analítica, como perceptiva. Este sistema automático también posibilita la aparición de aplicaciones totalmente nuevas, desde herramientas que pueden ayudar a cualquier persona a obtener exactamente la música que precisa, a usos radicalmente nuevos como la música adaptativa, aplicada a terapia, diversión o cualquier otro propósito. Para nosotros queda claro que aún queda mucho trabajo por hacer en este campo y que surgirán innumerables e inimaginables usos derivados de él.

Creatividad computacional

La creatividad es un concepto relativamente reciente, antes de la edad moderna la idea se identificaba más con inspiración divina que con algo con origen exclusivamente humano. Desde entonces se ha considerado una de sus características distintivas, lo cual implica por un lado cierto recelo o al menos escepticismo al atribuir creatividad a otras entidades, pero por otra parte, tratar de construir máquinas con esta cualidad representa un reto para la ciencia e ingeniería. En el desarrollo de la informática se ha tratado explícitamente de alcanzar la creatividad en el campo de las artes, normalmente el más asociado a esta habilidad. Sin embargo también se ha buscado en muchas otras áreas; un caso reciente conocido es el programa Watson, de IBM, capaz de procesar lenguaje natural, extraer información, representarla, razonar, aprender sobre ella y producir resultados originales. Este sistema se empleó en el concurso televisivo Jeopardy!, donde derrotó a sus oponentes humanos. Posteriormente se usó en otros campos, incluyendo aplicaciones financieras, problemas médicos, consultoría legal o incluso cocina. También se ha observado un creciente interés en el estudio de la creatividad, con algunas propuestas para medirla y para distinguir sistemas creativos de otros que simplemente tratan de aparentarlo (Colton, López de Mantaras, & Stock, 2009) (Bringsjord, Bello, & Ferrucci, 2003) (Colton, 2008) (Pease & Colton, 2011).

En inteligencia artificial se han usado diversas técnicas para lograr creatividad en muchos campos. El sistema que proponemos incorpora aspectos principalmente de tres de ellos: (1) Sistemas basados en conocimiento, normalmente implementados en forma de reglas para un determinado dominio. Este enfoque se ha usado previamente en otros trabajos en música (Schaffer & McGee, 1997). (2) Gramáticas formales. Un vocabulario simple, con un proceso de reescritura que conduce a un producto final. Este tipo de técnicas se han empleado previamente, por ejemplo en diseño (Stiny, 1980) (Shea & Fenves, 1997). (3) Métodos de búsqueda, que incluyen en muchos casos el uso de heurísticos (Russell & Norvig, 2009) (Campbell, Hoane, & Hsu, 2002).

Nuestro sistema implementará una representación genética para las composiciones basada en gramáticas formales y se efectuará una búsqueda evolutiva, con evaluación y selección de las composiciones más adecuadas, según reglas o criterios globales y específicos construidos en base a un conocimiento proporcionado por expertos. La contribución primordial dentro del campo de la composición algorítmica será desarrollar un método capaz de producir composiciones musicales genuinas, presentando además un método que evalúa la creatividad aportada por el sistema.

Música por ordenador y composición algorítmica

El término música por ordenador se usa para referirse a toda la música que ha sido producida empleando medios informáticos. Recientemente han surgido nuevas herramientas para ayudar en el proceso de composición: para grabar y reproducir sonidos, mezclar y editar el audio, escribir partituras, sintetizar audio, etc. También existen herramientas para ayudar en los procesos creativos de composición, llamadas CAAC (Computer-Aided Algorithmic Composition), como SuperCollider (McCartney, 2002) o MAX/MSP (Puckette, 2002). Se han inventado nuevos instrumentos musicales, tecnologías de instrumentos virtuales y formatos para almacenar la música de forma simbólica (MIDI o MusicXML) o de forma explícita como onda de audio WAV, FLAC o MP3). Además han aparecido nuevos estilos musicales y algunos métodos para producir composiciones de forma automática.

Algunos trabajos formales en composición algorítmica empezaron a aparecer en la década de 1950, por ejemplo un trabajo no publicado de Caplin y Prinz en 1955 (Ariza, 2011), con un generador de líneas melódicas; la conocida y citada Suite de Hiller e Isaacson (Hiller Jr & Isaacson, 1957), basada en cadenas de

Markov y un sistema de reglas; MUSICOMP de Baker (Ames, 1987), los algoritmos estocásticos de Xenakis (Ames, 1987), el ordenador dedicado capaz de componer nuevas melodías relacionadas con otras que se proporcionan como entrada, empleando procesos de Markov (Olson & Belar, 1961); el algoritmo de Gill en 1963, que empleaba técnicas de IA clásica como búsqueda jerárquica con backtracking (Gill, 1963); PROJECT1 de Koenig (Ames, 1987), que usaba diversas técnicas, como la composición serial; o el framework de Padber, que empleaba técnicas procedurales (Padberg, 1964). Las máquinas empezaron a ser cada vez más económicas y potentes, por ello el campo de la composición algorítmica ha crecido; sin embargo, ha sido un proceso lento, con poco contacto entre científicos y artistas, con la mayoría de las iniciativas llevadas a cabo por estos últimos y con muy poca continuidad en las investigaciones. Salvo pocas excepciones, como CHORAL (Ebcioglu, 1988), EMI (Cope, 1992) o Melomics (Sánchez, Moreno, Albarracin, Fernandez, & Vico, 2013), desarrollado en esta tesis, la composición por ordenador no ha tenido gran relevancia y su uso tampoco se ha extendido en la sociedad.

Inspiración en la biología

La biomimética es un campo de la ingeniería consistente en desarrollar soluciones con inspiración en estructuras o comportamientos observados en la naturaleza. A lo largo de los años, este enfoque ha mostrado un gran potencial para resolver problemas informáticos complejos en multitud de campos, como la inversión financiera, la visión por computador, el tráfico en redes informáticas, el tráfico de la vía pública, los sistemas de control en electrodomésticos, el diseño industrial creativo o el arte. Algunos ejemplos conocidos con enfoque bioinspirado son las redes neuronales artificiales, la lógica difusa o la computación evolutiva. Los algoritmos bioinspirados se usan frecuentemente porque suelen producir soluciones creativas o imprevistas, que pueden no ser complemente óptimas, pero resuelven razonablemente bien el problema.

La evolución es uno de los principales mecanismos que ha contribuido a la diversidad y complejidad observada en la naturaleza. Estos mecanismos han inspirado la aparición de los algoritmos evolutivos, una metodología basada en una población de soluciones que se refina mediante un proceso iterativo consistente en evaluación de los individuos, selección de las mejores soluciones y producción de la siguiente generación de la población. En biología, el proceso de transformación de la primera célula de un organismo, llamada cigoto, en un organismo multicelular completo mediante un proceso de desarrollo regulado por la expresión selectiva de genes (Carroll, Grenier, & Weatherbee, 2004) (Forgács & Newman, 2005) (Mayr, 1961), juega un papel importante en el proceso de evolución (Marcus, 2003) (Lewontin, 2000) (Müller G. B., 2007) (Bateson, 2001) (Borenstein & Krakauer, 2008). La biología evolutiva del desarrollo (conocida como evo-devo) (Carroll S. B., 2005) estudia la evolución de los procesos de desarrollo, que en ciencias de la computación ha inspirado métodos análogos. En los algoritmos evolutivos, si se usa una codificación indirecta adecuada (Stanley & Miikkulainen, 2003), un pequeño genotipo puede desarrollarse en un fenotipo grande y complejo, y pequeñas variaciones en el código genético podrían conducir a grandes variaciones en el fenotipo. Esto supone un enfoque más escalable y robusto que una codificación directa y además cuenta con un gran potencial para proporcionar soluciones más originales. Algunos ejemplos encontrados que han seguido este tipo de metodologías pueden ser el diseño de antenas de satélites de la NASA (Hornby, Lohn, & Linden, 2011), el diseño de fibra óptica microestructurada (Manos, Large, & Poladian, 2007), la generación automática de juegos de mesa (Browne, 2008) o nuevas técnicas de animación de personajes (Lobo, Fernández, & Vico, 2012). El desarrollo embrionario se puede tratar, no solo como un proceso aislado dirigido por el genoma, sino en el que también puede influir un entorno natural (Forgács & Newman, 2005). De esta forma, la aparición de diversidad en las soluciones también dependerá de la interacción con el contexto físico durante el proceso de desarrollo. En ciencias de la computación se han desarrollado algunos trabajos inspirados en ello (Lobo D. , 2010) (Fernández, 2012) (Sánchez-Quintana, 2014).

Para desarrollar el sistema de composición musical propuesto, tomaremos ideas de esta disciplina bioinspirada, incluyendo representación genética para las composiciones con una codificación indirecta o implícita, que sufrirán un proceso de desarrollo complejo, afectado por un contexto físico, para dar lugar a la composición musical expresada de forma explícita.

Recuperación de información musical (MIR)

La recuperación de información musical (MIR, por sus siglas en inglés) es un campo interdisciplinar que consiste en obtener información de diferentes tipos de datos musicales. El ejemplo más usual es la obtención de información acerca del estilo, género, instrumentos o voces participantes, nivel de activación, valencia, ritmos, etc. a partir de archivos de audio. No obstante, las fuentes de datos sujetas al estudio con MIR comprenden cualquier tipo de formato, como los archivos de música simbólicos (por ejemplo en forma de partitura) o incluso información cultural relativa a las piezas musicales, que se puede encontrar en sitios web o redes sociales. Algunos de los propósitos más conocidos de MIR son: la construcción de sistemas de recomendación de música; sistemas de transcripción, para convertir audio en notación de partitura; herramientas de reconocimiento de canciones; o herramientas de categorización. En la actualidad, la mayoría de los servicios comerciales de música (Spotify, vevo, rdio, PANDORA, Shazam…), incluyen como parte de su funcionamiento, algún tipo de técnica MIR.

En investigación también ha surgido un interés creciente por MIR, existiendo conferencias internacionales dedicadas a su estudio como ISMIR y multitud de publicaciones; por citar algunos ejemplos: la tesis del Dr. Cory McKay (McKay, 2010), el algoritmo de Shazam (Wang, 2013) o técnicas basadas en información cultural (Lamere, 2008) (Schedl, Gómez, & Urbano, 2014).

D.1.2 Método de composición algorítmica

Melomics es un sistema de composición musical que basa su funcionamiento en una representación genética para las composiciones y un proceso de desarrollo encargado de interpretar estos códigos y expresar la música de una forma explícita en diversos formatos. Asimismo el sistema incorpora multitud de reglas que permiten evaluar la bondad de los resultados en distintos niveles del proceso de desarrollo, permitiendo filtrar aquellos resultados que no se ajusten a lo esperado y almacenar estructuras genéticas para su posterior uso, que sí cumplan de forma adecuada los requisitos.

El sistema se ha diseñado con dos versiones, una destinada a producir música atonal, caracterizada por la ausencia de reglas compositivas estrictas, especialmente reglas de armonía; y una segunda versión derivada, orientada a crear composiciones tonales, priorizando el tratamiento de la armonía, mediante las reglas empleadas en la teoría musical actual. En ambas versiones la representación genética tiene estructura de gramática formal y poseen un flujo de ejecución con pequeñas diferencias, principalmente ocasionadas por las diferencias al evaluar la bondad de los resultados producidos. En el resto de aspectos, ambos sistemas han sido diseñados siguiendo los mismos principios, ya que las expectativas globales son similares, se persiguen soluciones originales, útiles, mutables, etc.

Para dar soporte al resultado del desarrollo de un genotipo dentro el sistema, fue necesaria la definición de un formato de representación interna capaz de almacenar toda la información compositiva y de ejecución musical, con mayor capacidad expresiva que los estándares de música simbólica conocidos, como MIDI o MusicXML. El formato interno posee, de forma general, una estructura matricial de tres dimensiones: pista instrumental, tiempo y propiedad musical de nota (inicio, duración, tono, dinámica, efectos...). Adicionalmente se incluye información global (nombre de la composición, fecha de creación, estilo que se usó para producirla, parámetros de mezcla...) e información de pista (ID de rol, nombre del instrumento, parámetros de mezcla...). A partir de este formato, empleando las herramientas y los scripts apropiados, se pueden generar los distintos archivos de salida requeridos; en representación simbólica, como una partitura; o en forma de audio, como un archivo WAV.

Melomics, una tecnología que ha estado en continuo desarrollo durante más de seis años, puede considerarse una contribución notable en el campo de la composición algorítmica, habiendo alcanzado una gran cantidad de hitos durante este tiempo que, de forma resumida y en orden cronológico, enumeramos a continuación: en 2010 la Universidad de Málaga concede un premio spin-off a una propuesta de start-up basada en una aplicación de Melomics; la primera obra musical, Opus #1 es creada; el Ministerio de Ciencia e Innovación financia el proyecto de investigación MELOMICS; en 2011 se estrena en un concierto en directo Hello World!, una obra completa para orquesta de cámara, en el festival Keroxen de Tenerife; Gustavo Díaz-Jerez publica un artículo sobre la composición con el sistema Melomics en Leonardo Music Journal (Diaz-Jerez, 2011); en 2012 músicos reconocidos mundialmente, entre ellos la Orquesta Sinfónica de Londres graban distintas obras de cámara y sinfónicas que formarán parte del álbum Iamus; se celebra un concierto en directo en la ETSI Informática de la Universidad de Málaga, interpretando música de Melomics, como conmemoración del centenario del nacimiento de Alan Turing; la tecnología despierta gran atención de los medios, apareciendo en Discover Magazine entre los 100 hitos científicos del año (Berger, 2013); en 2013 se celebra el Melomics Hackathon en la Universidad de Berkeley; Melomics se presenta en Googleplex durante la sesión de SciFoo music-as-it-could-be; se emplea la tecnología en aplicaciones y experimentos y la aplicación eMTCP es recomendada en el portal de la American Chronic Pain Association; en 2014 se estrena el segundo álbum, 0music, con música perteneciente a géneros populares; se ejecuta un ensayo enfrentando la música de Melomics con música compuesta por humanos en el Museo Interactivo de la Música de Málaga y en el Conservatorio Superior de Música de Málaga; en 2015 se presenta Melomics en Seúl, en el evento SDF2015: Conscious Curiosity; y en marzo de 2016 se estrenan cuatro nuevas obras en MaerzMusic, un festival de música celebrado en Berlin.

Modelo atonal

La primera versión del sistema Melomics se diseñó fundamentalmente para la producción de música atonal, concretamente en el estilo clásico contemporáneo. Para su construcción se tuvieron en cuenta los elementos de la teoría musical occidental moderna, evitando sin embargo construcciones convencionales de cualquier estilo conocido, de forma que las características rítmicas, armónicas o estructurales no quedaran restringidas de partida por la música producida hasta la fecha.

El código genético en este sistema, almacenado en un archivo de texto plano con extensión ".gen", está basado en una gramática formal de tipo L-system, varios parámetros adicionales que sirven para guiar el proceso de reescritura de la gramática y otros valores que se emplearán en el proceso de traducción de la cadena resultante en elementos musicales, entre ellos se encuentran la duración de referencia, la escala musical para cada pista, el identificador de instrumento y su tesitura. En este modelo, los símbolos que no poseen una regla de reescritura explícita, se consideran los genes operadores, cuya misión será modificar los valores de un parámetro musical (tono, duración, tiempo de inicio de nota, efectos, volumen, tempo…) durante el proceso de traducción de la cadena de símbolos a su forma más explícita en forma de matriz numérica. Por otra parte, los símbolos con reglas de producción explícitas pueden representar, tanto unidades estructurales musicales, como las notas musicales que finalmente serán ejecutadas, dependiendo de cómo suceda el proceso de reescritura, según los parámetros de iteración para la gramática y la función que se ha definido para computar las iteraciones restantes de cada regla. En cualquier caso, estos últimos símbolos descritos tendrán un identificador de instrumento asociado.

La cadena de símbolos resultante de la reescritura sufrirá un proceso de estabilización adicional antes de ser interpretada en música. Algunas de estas operaciones consisten en la eliminación de código genético superfluo, como la aparición consecutiva de dos genes operadores idempotentes. Otros de estos reajustes estarán provocados por el contexto físico, por ejemplo, dependiendo del estilo musical en el que se desarrolle la composición o las características particulares de los instrumentos elegidos, puede ser necesario reescribir subcadenas para satisfacer restricciones impuestas por estos, tales como máximo número de notas consecutivas o la aparición o supresión de determinados efectos musicales.

La última etapa del proceso morfogenético consiste en interpretar la cadena de caracteres en una estructura con forma de matriz numérica que contendrá los valores musicales explícitos de cada nota de cada canal. El proceso se basa en una lectura de la cadena de forma secuencial de izquierda a derecha, donde el comportamiento general consiste en que cada vez que aparece un gen operador, desplazará el valor actual de la variable musical que tiene asociada, partiendo de un valor inicial por defecto o indicado en el código genético. Otros operadores simplemente establecerán un valor específico para las variables. Cuando un gen instrumental aparece, se genera una nota musical adquiriendo el valor actual de cada uno de los parámetros musicales. A partir de la información que quede en esta estructura, el formato de representación interno que se ha diseñado, será posible obtener distintos formatos de representación de música estándares, tanto simbólicos (Moreno-Arcas, 2011), como sintetizados (De Vicente Milans, 2013).

El conocimiento musical experto es empleado, por una parte, como guía en el proceso de creación de genotipos, limitando las opciones en los procesos estocásticos, por ejemplo, con restricciones según la física de un instrumento, impidiendo que se generen más notas simultáneas de las que es capaz de ejecutar. Por otra parte el conocimiento se usa para la construcción de las funciones de fitness de este sistema evolutivo, normalmente incluyendo reglas estéticas poco restrictivas, relativas a la rítmica, la armonía o los contornos melódicos.

Bien para probar la capacidad del sistema o bien para producir resultados con un fin específico, hasta la fecha se han efectuado numerosos experimentos con el sistema compositivo, donde podemos destacar los siguientes hitos:

Iamus Opus #1. Esta obra puede considerase la primera pieza profesional de música clásica contemporánea compuesta por un ordenador en su propio estilo. Fue creada en 2010, tras finalizar el desarrollo de la versión primaria del sistema atonal, en colaboración con Gustavo Díaz Jerez.

Hello World!. Estrenada en 2011, en el festival Keroxen en Santa Cruz de Tenerife, es considerada la primera composición completa totalmente creada por un ordenador, usando notación musical convencional. En esta segunda iteración de Melomics, se disponía de mayor capacidad para gestionar instrumentos orquestales, se disponía de nuevas reglas compositivas, la escritura a formato MusicXML estaba completa y se había mejorado el sistema para producir composiciones de mayor duración y conjunto instrumental.

Ingeniería genética: Nokia tune. Con el fin de probar algunas de las ventajas que puede proporcionar el disponer de una composición en su representación genética, se llevó a cabo un ejercicio de ingeniería inversa, diseñando un genotipo del tema musical conocido como Nokia tune. A continuación se implementaron distintas funciones de mutación genética y un mecanismo de cruzamiento, para efectuar ingeniería genética automatizada. La ejecución de una batería de pruebas con estas funciones sirvió para estudiar las relaciones entre los diferentes descendientes del tema original, mediante un análisis de la música resultante.

Repositorio web y álbum Iamus. El sistema usado para producir Hello World! fue mejorado para gestionar una mayor diversidad instrumental y se implementó una primera versión de un interfaz de definición de estilos musicales. Como resultado se produjeron decenas de miles de obras, con multitud de configuraciones en los parámetros de entrada, que fueron usadas para poblar un repositorio musical accesible a través de la web. De esta generación se seleccionaron algunas obras y se grabaron en estudio, para dar lugar al álbum Iamus que se lanzó en 2012.

Música para terapias de relajación. La herramienta de especificación de estilos para el sistema atonal se usó para producir distintos tipos de música, concretamente con las configuraciones requeridas para las aplicaciones propuestas de música adaptativa.

Modelo tonal

El modelo diseñado para generar música atonal permite representar cualquier tipo de música, sin embargo para restringir la producción de composiciones a géneros más populares y obtenerlas en un tiempo de ejecución menor, fue necesario adaptar el sistema con el fin de permitir una representación más simple de los elementos usuales en música tonal (modos y progresiones armónicas, modos y patrones rítmicos, relaciones instrumentales muy definidas…). Se conservó el diseño global del sistema. Se añadieron nuevos símbolos en el código genético, principalmente para la gestión de la armonía y la estructura compositiva, haciéndose esta última corresponder con la organización jerárquica que se describe en teoría musical (Bent & Pople, 2015). La interpretación de los nuevos símbolos consta de un mayor nivel de abstracción, siendo por tanto el proceso de traducción más complejo. Por último se añadieron nuevos parámetros al código genético para guiar el proceso de desarrollo de la gramática y la traducción de la cadena final en la matriz musical

Para facilitar la interacción con el sistema de música tonal y permitir la especificación sencilla de reglas y dar lugar a estilos musicales concretos, desarrollamos una herramienta con forma de cuestionario sobre elementos musicales, expresados en un lenguaje de composición de un alto nivel de abstracción. Se programó un componente para traducir esta información, en restricciones para la generación de genotipos, restricciones en desarrollo y reglas de evaluación del fitness. De esta forma, sin necesidad de conocer el funcionamiento del sistema, simplemente disponiendo de conocimientos sobre composición, es posible lograr que el sistema produzca contenido musical de una forma dirigida.

El modelo genético para el sistema tonal puede ser descrito como una gramática formal determinista de contexto libre, donde los símbolos no terminales se identifican con elementos estructurales de una composición musical, como la composición en sí, períodos, frases o ideas y los elementos terminales se corresponden con manipuladores de las propiedades musicales como el pitch, duración, acorde actual, etc. y con las notas musicales en sí, que toman esas propiedades.

La estructuración jerárquica que se suele usar para construir genomas musicales con este modelo consta de 5 niveles: (1) composición, que está compuesta de una secuencia de (2) períodos, los cuales se desarrollan en un conjunto de (3) frases, que están formadas por (4) ideas, las cuales constan finalmente de una secuencia de (5) notas. A pesar de esta restricción se ha procurado mantener, e incluso mejorar algunas de las ventajas que aporta el modelo genético gramatical, como la legibilidad, la flexibilidad, la gran capacidad expresiva y la robustez ante alteración de símbolos y reglas de producción.

La primera etapa del proceso de transcripción es desarrollar la gramática de contexto libre hasta sustituir todos los símbolos no terminales en terminales, normalmente tras cuatro iteraciones, correspondientes a la estructura musical jerárquica que se ha descrito. Al igual que en el modelo atonal, la cadena resultante sufrirá procesos de ajustes y estabilización, algunos para eliminar elementos superfluos y otros son operaciones indicadas por determinados reguladores genéticos, como la adición, supresión o alteración de determinados símbolos genéticos o una secuencia de ellos, que pueden depender del estilo musical.

La segunda etapa del desarrollo de un genoma musical es la interpretación de la cadena para dar lugar a una estructura matricial con los valores explícitos de cada nota. Igual que en el modelo atonal, el proceso consiste en una lectura secuencial de izquierda a derecha con un funcionamiento general basado en desplazar los valores musicales actuales de los distintos parámetros, mediante los operadores correspondientes y cada vez que aparece un símbolo terminal asociado a un instrumento, se crea una nota que adquiere las propiedades actuales. Estos mecanismos son más complejos que en el sistema atonal, pudiendo estar influidos por el estilo musical. Por ejemplo, para computar el pitch concreto de una nota musical, es necesario tener en cuenta diversos parámetros como el acorde actual, la nota base del acorde, desplazamientos previos de tono o dentro de la escala e incluso los acordes permitidos para el rol tratado.

Algunas de las pruebas realizadas con música tonal para probar la capacidad del sistema o para producir resultados específicos incluyen:

Ingeniería genética: SBS Logo. Como ejemplo de aplicación comercial se hizo un experimento análogo al realizado con el Nokia tune en el sistema atonal. El sistema Melomics se presentó en el evento SDF2015 en Seúl, que fue retransmitido por el canal privado SBS. Por ello se tomó el tono corporativo de la empresa, se efectuó ingeniería inversa con la ayuda de un experto para obtener un código genético en el sistema tonal y se produjo un árbol genealógico con el logo original como el familiar central, ilustrando (1) un posible proceso evolutivo que podría haber dado lugar a este tono, (2) diferentes mutaciones del logo y (3) contaminación del código genético de un tema musical independiente con material genético de este tono, preservando la identidad del tema original, pero siendo posible identificar el SBS Logo como parte de él.

Aplicación para dolor crónico. Se continuó con el trabajo iniciado con el modelo anterior de producir música para relajación. La principal novedad consistió en el empleo de hypersongs, formados por un conjunto determinado de versiones de una misma composición, que son usadas de forma selectiva dentro del sistema desarrollado de música adaptativa, que se detallará más adelante.

Repositorio web y álbum 0music. A inicios de 2014 se implementaron numerosas configuraciones de estilos para el sistema tonal, dando lugar a otra colección en el repositorio de música, esta vez de estilos más populares. Una vez más, se seleccionaron algunos de estos temas y se publicaron en forma de álbum, conocido este como 0music.

Estilos DocumentaryPop y DocumentarySymphonic. El sistema tonal se configuró a través de la herramienta de estilos para generar música de dos tipos, que se usarían en documentales y otro tipo de producciones. En 2014 se produjeron 200 composiciones en estos estilos que se entregaron a una empresa de distribución musical.

D.1.3 Propiedades de la música generada

La música generada por Melomics se ha analizado desde una perspectiva analítica, empleando las herramientas usuales para extraer información musical, a partir de archivos de audio y de notación simbólica. También se ha hecho un estudio desde una perspectiva perceptiva, enfrentando música de Melomics con composiciones hechas por músicos al juicio de evaluadores humanos.

Análisis con MIR

Este tipo de estudio tenía como objetivo, por una parte comprobar la relación entre la codificación genética de las composiciones y la música resultante, concretamente cómo afectan los cambios en el código genético en los valores musicales finales, o si códigos genéticos producidos con una configuración de estilo concreta, realmente dan lugar a composiciones agrupadas en un estilo musical y además estas difieren de las producidas con otro estilo. Por otra parte se trató de comprobar analíticamente las características de la música producida con Melomics en relación a la música existente compuesta por humanos y en relación a otros tipos de sonidos.

En un primer análisis se usó la herramienta jSymbolic para extraer, de cada composición, los valores de 56 características musicales diferentes, incluyendo parámetros relacionados con el ritmo, textura, dinámica, pitch, melodía y armonía. El primer conjunto analizado fue una colección con aproximadamente 1000 variaciones del Nokia tune, producidas con distintos operadores genéticos implementados, donde se observó que la distancia, medida dentro de este espacio de parámetros con respecto al tono original, se incrementaba con la disrupción genética introducida. En el modelo tonal se llevó a cabo un experimento similar con el árbol filogenético del SBS Logo.

Empleando la misma herramienta se analizaron 656 obras de cámara del sistema atonal, con distintos conjuntos de instrumentos, en el estilo clásico contemporáneo. Tras calcularse el centroide de este grupo, se analizaron nuevas piezas pertenecientes a la misma configuración de estilo, las cuales se ubicaron, sin sorpresas, esparcidas entre las obras del estilo de referencia; a continuación se analizaron piezas pertenecientes a un etilo etiquetado como "Modern Classical" de una colección de composiciones en MIDI creadas por humanos, denominado *Bodhidharma*, y estas aparecieron en su mayoría ubicadas entre las composiciones del estilo clásico contemporáneo de Melomics, indicando que existe solapamiento entre ambos estilos musicales; por último se analizaron piezas producidas por el sistema tonal en el estilo "Disco02", las cuales aparecieron en el espacio de parámetros musicales, más alejadas del estilo de referencia que el resto. En un estudio similar se comprobó que el parecido musical entre los resultados de los estilos configurados "Disco02" y "DancePop" (evolución del primero) era mayor que entre "DancePop" y "DocumentarySymphonic" (estilo configurado independientemente).

Por último se trató de poner en contexto los diferentes estilos de música de Melomics, empleando las herramientas que proporciona la plataforma The Echo Nest. Entre sus denominados atributos acústicos, "Valence" parece aportar información útil para distinguir contenido musical de otro tipo de sonidos e incluso diferenciar la música más popular, de la música contemporánea, que en general carece de configuraciones armónicas comunes (debido a que probablemente incluya un análisis de la armonía, para puntuar la emoción evocada).

Estudio perceptivo

En 2014, se llevó a cabo un estudio propuesto en colaboración con el Dr. Alfredo Raglio y efectuado en el Museo Interactivo de la Música de Málaga y en el Conservatorio Superior de Música de Málaga, siguiendo la idea del Musical Output Toy Test (MOtT) (Ariza, 2009), inspirada en el Test de Turing, pero sustituyendo el papel del interrogador por cuestionarios para que los participantes o críticos, emitan su juicio sobre las piezas que se les presentan. El objetivo final del experimento fue medir la valencia de las composiciones, la capacidad de composición y la capacidad de interpretación de Melomics en comparación a la humana. Para ello se crearon dos piezas musicales con las mismas especificaciones de estilo, balada de guitarra, una por Melomics y otra por un compositor humano. Posteriormente ambas piezas fueron interpretadas por músicos humanos y por el sistema de síntesis de Melomics. Las cuatro piezas resultantes y una grabación de duración similar, conteniendo sonidos naturales (The sounds of Nature Collection (Jungle River, Jungle birdsong and Showers), 2010) combinados con sonidos de animales (Animals/Insects, n.d.), se emplearon para elaborar una encuesta con preguntas sobre emociones evocadas, imágenes mentales producidas y origen compositivo e interpretativo de las piezas musicales. Los participantes del experimento se clasificaron en dos grupos, de acuerdo a su experiencia en música, con más de cinco años se consideraron como músicos y en otro caso como no-músicos. Tras el análisis de los datos aportados por los 251 sujetos totales, no se observa sesgo en cuanto a la atribución del origen compositivo de las piezas musicales, y esta imposibilidad de discriminar se dio en ambos grupos por separado, músicos y no-músicos.

D.1.4 Composición musical adaptativa

Disponer de un sistema compositivo automático, además de la posibilidad de producir gran cantidad de composiciones musicales, permite otra serie de ventajas adicionales, como la fácil obtención de versiones de piezas musicales compuestas previamente o la generación rápida y sin fallos de piezas que satisfagan especificaciones musicales concretas. Una aplicación original de esta tecnología es la música adaptativa, que consiste en proveer música que cambia en tiempo real, en respuesta al valor detectado de determinadas variables en el oyente o en su entorno. En la literatura se han encontrado diversas aplicaciones de este concepto. Sin embargo, observamos que se usan bien composiciones previamente producidas por artistas humanos, que al cambiar la entrada detectada, se hace alternar la reproducción entre unas piezas y otras (en ocasiones con un simple efecto crossfade); o se usan señales de audio generadas en directo, pero que distan de ser composiciones musicales reales.

Nuestra contribución en esta área es un método capaz de proveer con un flujo de música genuina que se adapta de forma continua y lo hace mediante la manipulación de los parámetros musicales de una composición en los distintos niveles de abstracción; incluyendo variables como el ritmo, la evolución de la armonía, de la densidad instrumental, relación entre roles instrumentales o la estructura compositiva.

Sistema de generación basado en realimentación. Hypersong

Se ha implementado un sistema estandarizado para dar soporte al concepto de composición adaptativa en tiempo real para las aplicaciones propuestas, el cual requirió los siguientes elementos:

- En cada aplicación, un componente para evaluar el estado del entorno, por ejemplo el estado de ansiedad de un sujeto mediante el análisis de su frecuencia cardíaca, que será la forma de definir cómo debe ser el flujo de música que se proporciona.
- Un mecanismo para traducir la información del análisis del entorno, en los parámetros para seleccionar la música apropiada de salida, implementado mediante una modificación del modelo de autómata de Moore.
- La definición del tipo de música a proporcionar en cada aplicación y estado, que se realiza a través de la herramienta de configuración de estilos. Se definió e implementó el concepto de hypersong, una estructura de audio musical de dos dimensiones: versión de la composición y tiempo (cada versión estará dividida en fragmentos de audio seccionados según la estructura compositiva). De esta forma, dada la información del autómata y el instante actual de reproducción, el sistema es capaz de indexar el fragmento temporal y versión de las hypersongs que ha de añadir a continuación en el flujo de reproducción normal.

Aplicaciones

Haciendo uso de la tecnología musical desarrollada, se diseñaron una gran variedad de aplicaciones, entre ellas destacamos:

eMTSD. Una aplicación para dispositivos móviles, para facilitar la iniciación del sueño. Funciona situando el dispositivo bajo la almohada, de forma que sea posible caracterizar el estado de somnolencia, a través del análisis de la actividad detectada con el acelerómetro en el usuario. Esta información se usa para alimentar un sencillo autómata de cuatro estados, que establece la reproducción de una versión de las composiciones progresivamente más suave, según se detecte un estado de somnolencia mayor, hasta detener la reproducción completamente cuando el sujeto está dormido. Posteriormente se desarrollaron versiones de esta aplicación, con funcionamiento y música ligeramente diferentes para su uso durante la siesta y para ayudar a la conciliación del sueño de niños pequeños.

PsMelody. Una aplicación de escritorio, implementada en Windows 7, para favorecer el rendimiento en el trabajo y al mismo tiempo tratar de disminuir los niveles de estrés. PsMelody analiza las pulsaciones del teclado y el movimiento del ratón para caracterizar el tipo de actividad que se está realizando y la energía que se emplea. A partir de ello, se reproducen composiciones que cambian de versión para favorecer el trabajo, pero evitando caer en situación de ansiedad.

@car**.** Una aplicación móvil para favorecer la concentración durante la conducción y tratar de reducir la tensión, especialmente bajo condiciones irregulares como tiempo atmosférico extremo o tráfico saturado. Como resultado de este trabajo se publicó un Proyecto de Fin de Carrera en 2013 (Gutiérrez-Gutiérrez, 2013). Esta aplicación toma como entrada los datos proporcionados por el acelerómetro, el GPS, el magnetómetro y el fotómetro. A partir de esta información computa el estado actual de la conducción, distinguiendo cuatro niveles de fatiga o dificultad, desde conducción con tráfico saturado y condiciones climáticas adversas, hasta conducción fluida por vía interurbana a alta velocidad. La versión de la música reproducida es más suave cuando la dificultad de la conducción es mayor.

@memory. Esta aplicación móvil hace uso de la música para ejercitar la memoria asociativa, proponiendo un juego basado en identificar melodías con figuras que se muestran en pantalla previamente (la música no es adaptativa en este caso) y en cada fase la dificultad irá incrementando. Tanto el funcionamiento como las especificaciones musicales de esta aplicación sirven como base para futuras propuestas para tratar enfermedades relacionadas con la memoria.

@rhythm. Una aplicación para ayudar a mejorar la concentración y la coordinación. Presenta pequeños trozos de música y se debe tratar de identificar el ritmo predominante mediante pulsaciones sobre la pantalla táctil del dispositivo en un lugar concreto. Igual que la anterior aplicación, hace uso del repositorio de melodías automáticamente generadas, versionadas y distribuidas por Melomics.

eMTCP. Es una aplicación móvil dirigida a disminuir los niveles de dolor en pacientes que sufren de dolor crónico, aprovechando el efecto distractor de la música. Se implementó para llevar a cabo un experimento clínico cuyas hipótesis eran que la música adaptativa mejora la calidad del sueño en pacientes con dolor crónico, correspondiéndose con una reducción de la percepción del dolor; y que la música adaptativa reduce la percepción subjetiva del dolor por parte del sujeto. Se implementó una máquina de estados que distinguía entre ocho estados de ansiedad, transitándose de uno a otro según el grado de dolor indicado por el usuario de forma interactiva.

STM. El Sistema de Terapia Musical para hospitales fue el primer sistema de música adaptativa implementado. Se desarrolló en lenguaje MATLAB para funcionar en pequeños ordenadores portátiles y cuenta con una interfaz gráfica de usuario sencilla. Se empleó para llevar a cabo experimentos de terapias de relajación, con diferentes configuraciones en pacientes de clínicas y hospitales. El primero de los ensayos propuestos consistió en un experimento de doble ciego donde se pretendía comprobar la mejora en la tasa de lactancia de recién nacidos, cuando se proporcionaba música relajante adaptativa a las madres para disminuir su nivel de ansiedad. El parámetro de entrada en este caso, para caracterizar el estado de agitación, es la frecuencia cardíaca, la cual se analiza teniendo en cuenta su desviación con respecto a la frecuencia basal, previamente calculada.

@jog. Una aplicación móvil que efectúa podometría usando el acelerómetro, para detectar el ritmo dominante actual de un corredor y reproducir música de acuerdo con esta información. El smartphone ha de ir situado en el brazo del corredor, la interfaz de usuario es sencilla para facilitar una posible manipulación durante el ejercicio y la aplicación posee distintas configuraciones y niveles de dificultad, diseñadas para mejorar el rendimiento deportivo. Las hypersongs empleadas en esta aplicación están formadas por 25 versiones, con distinto bpm como principal variación entre ellas, aunque no la única. El estilo de base es música disco, pero con una gran apertura de los parámetros, para abarcar una mayor cantidad de preferencias musicales, sin perder la capacidad de animar al usuario a mantener un ritmo.

@bike. Es una versión de la aplicación @jog para ayudar a mejorar el rendimiento en el ciclismo (de interior o exterior). Cuenta con una configuración de parámetros distinta para la detección del ritmo, en este caso el dispositivo ha de ir bien fijado en el muslo o en los gemelos del deportista. El estilo musical empleado es el mismo que en @jog, pero se aumenta el rango de bpm, para satisfacer la mayor cantidad de ritmos que pueden darse usando bicicleta.

D.2 Conclusiones

En esta tesis hemos presentado el Sistema de composición Melomics, describiendo el diseño de las dos versiones del algoritmo, ambas basadas en métodos bioinspirados, detallando los principales mecanismos implicados y la forma en la que se trata la información generada a lo largo del proceso. Hemos evaluado los resultados que se obtienen con este sistema empleando diferentes enfoques. Por último hemos expuesto una nueva forma de proveer música aprovechando las características de la composición automatizada.

Se han usado distintos métodos dentro del campo de la inteligencia computacional. Para la definición de la música y los estilos, hemos usado técnicas basadas en conocimiento, para proporcionar soluciones válidas hemos empleado búsqueda evolutiva y, como soporte para representar y manipular los elementos musicales siguiendo este enfoque, se ha usado una representación genética implícita con un proceso de desarrollo asociado. Hemos creado un método capaz de producir música genuina y tratarla; siendo éste escalable, lo que significa que parte de una pieza musical puede reutilizarse como parte de otra con una estructura más compleja; con una sintaxis muy expresiva, ya que ha permitido representar cualquier pieza musical, de la misma forma que podría hacerse empleando notación musical estándar; es flexible, una composición puede representarse de multitud de formas diferentes; es compacto, esto es, que a pesar de incluir toda la información compositiva y de interpretación, ocupa menos espacio de almacenamiento que cualquier otro formato simbólico conocido y por supuesto menos que cualquier formato de audio; y por último es robusto, lo que significa que un código genético alterado de cualquier forma, empleando los símbolos permitidos, da lugar a una pieza musical que no solo es válida, sino que además compartirá elementos compositivos con la pieza correspondiente al genotipo original.

Con este sistema hemos generado Iamus Opus #1, la que puede ser considerada como la primera pieza de música profesional en el estilo clásico contemporáneo, compuesto por un ordenador en su propio estilo. Posteriormente Hello World!, la primera composición completa compuesta por un ordenador y empleando notación musical convencional; un repositorio de música conteniendo diferentes estilos musicales y descargable en distintos formatos simbólicos y de audio, bajo licencia CC0; y dos álbumes musicales, Iamus album, con música contemporánea y 0music, con música de estilos más populares.

Hemos desarrollado una herramienta que permite especificar estilos musicales respondiendo a preguntas planteadas en forma de cuestionario, sin necesidad de conocer el funcionamiento del sistema, simplemente poseyendo conocimientos esenciales en composición musical.

Durante el desarrollo de este sistema hemos adquirido experiencia que se ha ido reflejando en sucesivos refinamientos y en la mejora de la especificación y tratamiento de parámetros, estableciendo las bases para un futuro método automático de búsqueda evolutiva alternativo. Hemos propuesto e implementado una nueva forma de representar música explícitamente en notación simbólica, incluyendo también directrices sobre la interpretación de la pieza y meta-información sobre el proceso compositivo. Adicionalmente se estableció un formato, basado en sintaxis JSON, para proporcionar información sobre las composiciones a sistemas externos.

El Test de Turing, la prueba de referencia para evaluar sistemas artificiales inteligentes, por ser interactivo, no es tan adecuado para evaluar los productos artísticos. Por ello se presentó un método diferente, donde el papel del interrogador se sustituye por críticos que emiten un juicio sobre los resultados. Se ejecutó un experimento tomando esta idea, donde se constató que, bajo las condiciones establecidas, no es posible distinguir la música compuesta por Melomics, de la creada por un compositor humano.

Por último hemos presentado el concepto de hypersong, una estructura musical que hace viable el empleo de la música adaptativa con nuestro sistema. Esta tecnología, que puede proporcionar un flujo continuo de música adaptado según criterios especificados, también se hace accesible públicamente a través de una API web, alimentado este sistema mediante un repositorio de música en formato hypersong y en formato estándar.

Además de los logros mencionados, esta investigación deja abiertas multitud de líneas de trabajo, tanto en forma de aplicaciones comerciales que exploten la tecnología existente, como en líneas de investigación. En este último caso identificamos tres direcciones inmediatas: (1) Un algoritmo genético, como alternativa al sistema actual, el cual solo salva parte del material genético para ser usado en un futuro, en lugar de los genotipos completos. Incorporar este método al sistema no requeriría gran esfuerzo de codificación, al haber orientado desde el principio el diseño en este sentido, incluyendo la representación genómica implícita. Para la función fitness se han estudiado diferentes propuestas, aplicables según el objetivo que se persiga. Por último, los operadores genéticos se desarrollarían partiendo de las funciones implementadas para los experimentos de ingeniería genética descritos. El principal objetivo en esta línea de trabajo sería la búsqueda de unas soluciones que igualmente satisfagan los requisitos especificados, pero siendo estas más inesperadas o creativas. (2) Alcance de nuevos estilos y géneros musicales. A pesar de la capacidad expresiva de sus formatos, Melomics actualmente solo produce música en un conjunto de estilos reducidos con respecto al total de la música. Sería necesaria una ampliación del conocimiento experto manejado por la herramienta de estilos, para dar soporte a la exploración del resto del espacio musical, tratando de alcanzar toda la música conocida primero y posteriormente nuevos géneros. (3) Música personal. Sería un avance al concepto de música adaptativa, consistiendo en un flujo de música que responde a cualquier contexto y que se adapta a las preferencias del usuario, dando lugar a un flujo musical único para cada persona. Un primer intento se hizo mediante una aplicación denominada @life, que recogía el funcionamiento de muchas de las aplicaciones adaptativas previas. Para lograr el objetivo sería necesario por un lado tener la capacidad de detectar la actividad del usuario y sus preferencias musicales de forma automática, y por otra parte abarcar una mayor cantidad de estilos en la generación de música.

Bibliography

Albarracín-Molina, D. D. (2010). Diseño e implementación de una estrategia para un juego de conexión. Master thesis. Universidad de Málaga.

Albarracin-Molina, D. D., Moya, J. C., & Vico, F. J. (2016). An evo-devo system for algorithmic composition that actually works. *Proceedings of the Companion Publication of GECCO 2016.* Denver, Colorado, USA.

Alfonseca, M., Cebrián Ramos, M., & Ortega, A. (2005). Evolving computer-generated music by means of the normalized compression distance. *WSEAS Transactions on Information Science and Applications*.

Ames, C. (1987). Automated composition in retrospect: 1956-1986. *Leonardo*, 169-185.

Animals/Insects. (n.d.). Retrieved 2015, from freeSFX: http://www.freesfx.co.uk/soundeffects/animals_insects/

Ariza, C. (2009). The interrogator as critic: The turing test and the evaluation of generative music systems. *Computer Music Journal, 33*(2), 48-70.

Ariza, C. (2011). Two pioneering projects from the early history of computer-aided algorithmic composition. *Computer Music Journal, 35*(3), 40-56.

Ball, P. (2012, July 1). Iamus, classical music's computer composer, live from Malaga. *The Guardian*. Retrieved from http://www.theguardian.com/music/2012/jul/01/iamus-computer-composes-classical-music

Ball, P. (2014, August 8). Artificial music: The computers that create melodies. *BBC*.

Barceló, A., Barbé, F., Llompart, E., De la Peña, M., Durán-Cantolla, J., Ladaria, A., . . . G., A. A. (2005). Neuropeptide Y and leptin in patients with obstructive sleep apnea syndrome: role of obesity. *American journal of respiratory and critical care medicine, 171*(2), 183-187.

Bateson, P. (2001). Where does our behaviour come from? *Journal of biosciences, 26*(5), 561-570.

Bent, I. D., & Pople, A. (2015). Analysis. In D. L. Root (Ed.), *Grove Music Online.* Oxford University Press.

Bentley, P. J. (1999). Is evolution creative. *Proceedings of the AISB*, *99*, pp. 28-34.

Bentley, P., & Corne, D. (2002). An introduction to creative evolutionary systems. In P. Bentley, & D. Corne, *Creative evolutionary systems.* Morgan Kaufmann.

Berger, K. (2013). 100 Top Stories of 2012. Digital Composer Records with London Symphony Orchestra. *Discover Magazine, 24*(1).

Biles, J. A. (1998). Interactive GenJam: Integrating real-time performance with a genetic algorithm. *Proceedings of the 1998 international computer music conference*, (pp. 232-235).

Boden, M. A. (2009). Computer models of creativity. *AI Magazine, 30*(3), 23.

Borenstein, E., & Krakauer, D. C. (2008). An end to endless forms: epistasis, phenotype distribution bias, and nonuniform evolution. *PLoS Comput Biol, 4*(10), e1000202-e1000202.

Boulanger, R. C. (2000). *The Csound book: perspectives in software synthesis, sound design, signal processing, and programming.* MIT press.

Bringsjord, S., Bello, P., & Ferrucci, D. (2003). Creativity, the Turing test, and the (better) Lovelace test. *The Turing Test* (pp. 215-239). Springer Netherlands.

Browne, C. (2008). Automatic generation and evaluation of recombination games. Doctoral dissertation. Queensland University of Technology.

Buttram, T. (2003). *DirectX 9 Audio Exposed: Interactive Audio Development, chap. Beyond Games: Bringing DirectMusic into the Living Room.* Wordware Publishing Inc.

Campbell, M., Hoane, A. J., & Hsu, F. H. (2002). Deep blue. *Artificial Intelligence, 134*(1-2), 57-83.

Carroll, S. B. (2005). *Endless forms most beautiful: The new science of evo devo and the making of the animal kingdom.* WW Norton & Company.

Carroll, S., Grenier, J., & Weatherbee, S. (2004). *From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design* (2 ed.). Wiley-Blackwell.

Chambel, T., Correia, L., Manzolli, J., Miguel, G. D., Henriques, N. A., & Correia, N. (2007). Creating video art with evolutionary algorithms. *Computers & Graphics, 31*(6), 837-847.

Christensen, T. (Ed.). (2002). *The Cambridge history of Western music theory.* Cambridge University Press.

Colton, S. (2008). Creativity Versus the Perception of Creativity in Computational Systems. *AAAI Spring Symposium: Creative Intelligent Systems*, (pp. 14-20).

Colton, S., & Wiggins, G. A. (2012). Computational creativity: the final frontier? *ECAI*, (pp. 21-26).

Colton, S., López de Mantaras, R., & Stock, O. (2009). Computational creativity: Coming of age. *AI Magazine, 30*(3), 11-14.

Colton, S., Pease, A., & Charnley, J. (2011). Computational creativity theory: The FACE and IDEA descriptive models. *Proceedings of the Second International Conference on Computational Creativity*, (pp. 90-95).

Cook, M., & Colton, S. (2011). Automated collage generation–with more intent. *Proceedings of the Second International Conference on Computational Creativity.*

Cope, D. (1992). Computer modeling of musical intelligence in EMI. *Computer Music Journal, 16*(2), 69-83.

Cope, D. (2005). *Computer models of musical creativity.* Cambridge: MIT Press.

Crocker, R. L. (1966). *A history of musical style.* Courier Corporation.

Dacey, J. (1999). Concepts of Creativity: A History. In M. A. Runco, & S. R. Pritzer, *Encyclopedia of Creativity* (Vol. 1). Elsevier.

Davidson, E. H. (2010). *The regulatory genome: gene regulatory networks in development and evolution.* Academic Press.

Davidson, E. H., & Erwin, D. H. (2006). Gene regulatory networks and the evolution of animal body plans. *Science, 311*(5762), 796-800.

De Smedt, T. D. (2013). *Modeling Creativity: Case Studies in Python.* University Press Antwerp.

De Vicente Milans, R. (2013). Síntesis musical humanizada con notación musical simbólica. Master thesis. Malaga: Universidad de Málaga.

Delgado, M., Fajardo, W., & Molina-Solana, M. (2009). Inmamusys: Intelligent multiagent music system. *Expert Systems with Applications, 36*(3), 4574-4580.

Diaz-Jerez, G. (2011). Composing with Melomics: Delving into the computational world for musical inspiration. *Leonardo Music Journal, 21*, 13-14.

DiBona, C., Stone, M., & Cooper, D. (2005). *Open sources 2.0: The continuing evolution.* O'Reilly Media, Inc.

Ebcioglu, K. (1988). An expert system for harmonizing four-part chorales. *Computer Music Journal, 12*(3), 43-51.

EL DÍA. (2011, October 14). Keroxen aúna coreografía y música sobre el escenario de El Tanque. *EL DÍA*.

Europa-Press. (2012, July 1). La UMA diseña un ordenador que compone música clásica. *El Mundo*.

Farnell, A. (2007). An introduction to procedural audio and its application in computer games. *Audio Mostly Conference*, (pp. 1-31).

Fernández, J. D. (2012). The Evolution of Diversity in the Structure and Function of Artificial Organisms. Doctoral dissertation. Málaga: Universidad de Málaga.

Fernandez, J. D., & Vico, F. J. (2013). AI Methods in Algorithmic Composition: A Comprehensive Survey. *Journal of Artificial Intelligence Research, 48*, 513-582.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., . . . Welty, C. (2010). Building Watson: An Overview of the DeepQA Project. *AI magazine, 31*(3), 59-79.

Ferrucci, D., Levas, A., Bagchi, S., Gondek, D., & Mueller, E. T. (2013). Watson: beyond jeopardy! *Artificial Intelligence, 199*, 93-105.

Fiebelkorn, T. (2003). *Patent No. WO2003018097 A1.*

Forgács, G., & Newman, S. (2005). *Biological physics of the developing embryo.* Cambridge University Press.

Garay Acevedo, A. (2004). Fugue composition with counterpoint melody generation using genetic algorithms. *Proceedings of the Second international conference on Computer Music Modeling and Retrieval* (pp. 96-106). Springer-Verlag.

Garner, T., Grimshaw, M., & Nabi, D. A. (2010). A preliminary experiment to assess the fear value of preselected sound parameters in a survival horror game. *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound* (p. 10). ACM.

Gartland-Jones, A. (2002). Can a genetic algorithm think like a composer? *Generative Art.*

Gendreau, M., & Potvin, J. Y. (Eds.). (2010). *Handbook of metaheuristics.* Springer.

Gero, J. S. (2000). Computational models of innovative and creative design processes. *Technological Forecasting and Social Change, 64*(2-3), 183-196.

Gero, J. S., & Maher, M. L. (Eds.). (1993). *Modeling Creativity and Knowledge-Base Creative Design.* Psychology Press.

Gervás, P. (2009). Computational Approaches to Storytelling and Creativity. *AI Magazine, 30*(3), 49-62.

Gill, S. (1963). A Technique for the Composition of Music in a Computer. *The Computer Journal, 6*(2), 129-133.

Gjerdingen, R. (1988). Concrete musical knowledge and a computer program for species counterpoint. In *Explorations in Music, the Arts, and Ideas: Essays in Honor of Leonard B. Meyer* (pp. 199-228). Pendragon Press.

Goodwin, B. (2001). Living form in the making. In B. Goodwin, *How the leopard changed its spots: the evolution of complexity* (pp. 77-114). Princeton University Press.

Gutiérrez-Gutiérrez, J. D. (2013). Aplicación móvil para la reproducción de música adaptada en tiempo real a los distintos momentos, estados y eventos que se producen durante la conducción de un automóvil. Master Thesis. Málaga: Universidad de Málaga.

Harnad, S. (2000). Minds, machines and Turing: The Indistinguishability of Indistinguishables. *Journal of Logic, Language, and Information, 9*(4), 425-445.

Hiller Jr, L. A., & Isaacson, L. M. (1957). Musical composition with a high speed digital computer. *Audio Engineering Society Convention 9. Audio Engineering Society.*

Holland, J. H. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press.

Honda, T. (1998). *Washington, DC: U.S. Patent and Trademark Office Patent No. D403,013.*

Hood, L., & Galas, D. (2003). The digital code of DNA. *Nature, 421*(6921), 444-448.

Hornby, G. S., & Pollack, J. B. (2002). Creating high-level components with a generative representation for body-brain evolution. *Artificial life, 8*(3), 223-246.

Hornby, G., Lohn, J. D., & Linden, D. S. (2011). Computer-Automated Evolution of an X-Band Antenna for NASA's Space Technology Mission. *Evolutionary computation, 19*(1), 1-23.

Horowitz, M. D. (1995). Representing musical knowledge. Doctoral dissertation. Columbia University.

Huang, S. T., Good, M., & Zauszniewski, J. A. (2010). The effectiveness of music in relieving pain in cancer patients: a randomized controlled trial. *International journal of nursing studies, 47*(11), 1354-1362.

Institute of Medicine (US). (2011). Relieving Pain in America: A Blueprint for Transforming Prevention, Care, Education, and Research. National Academies Press.

Ioja, S. W., & Rennert, N. J. (2012). Relationship between sleep disorders and the risk for developing type 2 diabetes mellitus. *Postgraduate medicine, 124*(4), 119-129.

Jennings, K. E. (2010). Developing creativity: Artificial barriers in artificial intelligence. *Minds and Machines, 20*(4), 489-501.

Johnson, M., Tauritz, D. R., & Wilkerson, R. (2004). Evolutionary computation applied to melody generation. *Proceedings of the Artificial Neural Networks in Engineering (ANNIE) Conference.*

Juslin, P. N., & Sloboda, J. A. (2001). *Music and emotion: Theory and research.* Oxford University Press.

Kim, J., Lee, S., Kim, S., & Yoo, W. (2011). Music mood classification model based on arousal-valence values. *Advanced Communication Technology (ICACT), 2011 13th International Conference on* (pp. 292-295). IEEE.

Lai, H. L., & Good, M. (2005). Music improves sleep quality in older adults. *Journal of advanced nursing, 49*(3), 234-244.

Lamere, P. (2008). Social tagging and music information retrieval. *Journal of New Music Research, 37*(2), 101-114.

Levine, M., & Tjian, R. (2003). Transcription regulation and animal diversity. *Nature, 424*(6945), 147-151.

Lewontin, R. (2000). *The Triple Helix: Gene, Organism, and Environment.* Harvard University Press.

Lindenmayer, A. (1968). Mathematical models for cellular interaction in. *Journal of Theoretical Biology*, 280-315.

Lobo, D. (2010). Evolutionary development based on genetic regulatory models for behavior-finding. Doctoral dissertation. Málaga: Universidad de Málaga.

Lobo, D., Fernández, J. D., & Vico, F. J. (2012). Behavior-finding: morphogenetic designs shaped by function. *Morphogenetic Engineering* (pp. 441-472). Springer Berlin Heidelberg.

Lozano, L., Medaglia, A. L., & Velasco, N. (2009). Generation of Pop-Rock chord sequences using genetic algorithms and variable neighborhood search. *Applications of Evolutionary Computing* (pp. 573-578). Springer Berlin Heidelberg.

Malmi, E., Takala, P., Toivonen, H., Raiko, T., & Gionis, A. (2015). DopeLearning: A Computational Approach to Rap Lyrics Generation. *arXiv preprint arXiv:1505.04771*.

Manos, S., Large, M. C., & Poladian, L. (2007). Evolutionary design of single-mode microstructured polymer optical fibres using an artificial embryogeny representation. *Proceedings of the 9th annual conference* *companion on Genetic and evolutionary computation* (pp. 2549-2556). ACM.

Marcus, G. (2003). *The birth of the mind: How a tiny number of genes creates the complexities of human thought*. Basic Books.

Markoff, J. (2011, February 16). Computer Wins on 'Jeopardy!': Trivial, It's Not. *The New York Times*, p. A1.

Martín, G. M. (2010). Evolving complexity and similarity in an Artifcial Life framework based on formal language theory. Doctoral dissertation. Málaga: Universidad de Málaga.

Mason, S., & Saffle, M. (1994). L-Systems, melodies and musical structure. *Leonardo Music Journal, 4*, 31-38.

Mayr, E. (1961). Cause and effect in biology. *Science, 134*(3489), 1501-1506.

McCartney, J. (2002). Rethinking the computer music language: SuperCollider. *Computer Music Journal, 26*(4), 61-68.

McKay, C. (2004). Automatic genre classification of MIDI recordings. M.A. Thesis. McGill University.

McKay, C. (2010). Automatic Music Classification with jMIR. Doctoral dissertation. McGill University.

Meyer, L. B. (1989). *Style and music: Theory, history, and ideology.* University of Chicago Press.

Moffat, D. C., & Kelly, M. (2006). An investigation into people's bias against computational creativity in music composition. *Assessment, 13*, 11.

Moreno-Arcas, F. (2011). Desarrollo de un sistema de modelado de partituras musicales en notación profesional a partir de un generador de estructuras musicales. Master Thesis. Málaga: Universidad de Málaga.

Morreale, F., De Angeli, A., Masu, R., Rota, P., & Conci, N. (2014). Collaborative creativity: The Music Room. *Personal and Ubiquitous Computing, 18*(5), 1187-1199.

Morreale, F., Masu, R., & De Angeli, A. (2013). Robin: an algorithmic composer for interactive scenarios. *Proceedings of SMC.* Stockholm.

Müller, G. B. (2007). Evo-devo: extending the evolutionary synthesis. *Nature Reviews Genetics, 8*(12), 943-949.

Müller, M. (2002). Computer go. *Artificial Intelligence, 134*(1), 145-179.

National Sleep Foundation. (2000). Sleep in America poll.

Newman, S. (2003). From physics to development: the evolution of morphogenetic mechanisms. In G. B. Müller, & S. A. Newman, *Origination of organismal form: beyond the gene in developmental and evolutionary biology* (pp. 221-240).

Olson, H. F., & Belar, H. (1961). Aid to music composition employing a random probability system. *The Journal of the Acoustical Society of America, 33*(9), 1163-1170.

Pachet, F. (2002). Interacting with a musical learning system: The Continuator. *Proceedings of the International Conference on Music and Artificial Intelligence*, (pp. 103-108).

Pachet, F., & Cazaly, D. (2000). A taxonomy of musical genres. *RIAO*, (pp. 1238-1245).

Padberg, H. A. (1964). Computer-composed canon and free-fugue. Ph.D. thesis. Saint Louis University.

Papadopoulos, G., & Wiggins, G. (1999). AI methods for algorithmic composition: A survey, a critical view and future prospects. *AISB Symposium on Musical Creativity*, (pp. 110-117). Edinburgh, UK.

Pearce, M., & Wiggins, G. (2001). Towards a framework for the evaluation of machine. *Proceedings of the Symposium on Artificial Intelligence and Creativity*, (pp. 22-32).

Pease, A., & Colton, S. (2011). On impact and evaluation in computational creativity: A discussion of the Turing test and an alternative proposal. *Proceedings of the AISB symposium on AI and Philosophy.*

Peckham, M. (2013, 01 4). Finally, a Computer that Writes Contemporary Music Without Human Help. (Time, Ed.) *Time*. Retrieved from Time: http://techland.time.com/2013/01/04/finally-a-computer-that-writes-contemporary-music-without-human-help/

Peppard, P. E., Young, T., Palta, M., & Skatrud, J. (2000). Prospective study of the association between sleep-disordered breathing and hypertension. *New England Journal of Medicine, 342*(19), 1378-1384.

Perez-Bravo, J. (2014). Montaje y configuración de un cluster de computación para su uso con Torque y Maui. Master Thesis. Málaga: Universidad de Málaga.

Prusinkiewicz, P. (1986). Score generation with L-systems. *Proceedings of the International Computer Music Conference*, (pp. 455-457).

Prusinkiewicz, P., & Lindenmayer, A. (1990). *The algorithmic beauty of plants.* Springer-Verlag New York.

Puckette, M. (2002). Max at seventeen. *Computer Music Journal, 26*(4), 31-43.

Pudaruth, S., Amourdon, S., & Anseline, J. (2014). Automated generation of song lyrics using CFGs. *2014 Seventh International Conference on Contemporary Computing (IC3)* (pp. 613-616). IEEE.

Rao, M. (2012, July 2). London Symphony Orchestra Plays Computer-Written Composition (VIDEO). *The Huffington Post*.

Redacción Creativa. (14 de October de 2011). Keroxen ofrece este fin de semana una intensa programación de música y danza. *Creativa Canaria*.

Reddin, J., McDermott, J., & O'Neill, M. (2009). Elevated pitch: Automated grammatical evolution of short compositions. *Applications of Evolutionary Computing* (pp. 579-584). Springer Berlin Heidelberg.

Requena, G., Sánchez, C., Corzo-Higueras, J. L., Reyes-Alvarado, S., Rivas-Ruiz, F., Vico, F., & Raglio, A. (2014). Melomics music medicine (M3) to lessen pain perception during pediatric prick test procedure. *Pediatric Allergy and Immunology, 25*(7), 721-724.

Rer Nat Wiebkin, H. D. (1985). *Patent No. DE3338649 A1.*

Ritchie, G. (2007). Some empirical criteria for attributing creativity to a computer program. *Minds and Machines, 17*(1), 67-99.

Ritter, P. L., González, V. M., Laurent, D. D., & Lorig, K. R. (2006). Measurement of pain using the visual numeric scale. *The Journal of rheumatology, 33*(3), 574-580.

Rossum, D., & Joint, E. (1995). The SoundFont® 2.0 File Format.

Runco, M. A., & Albert, R. S. (2010). Creativity research: a historical view. In J. C. Kaufman, & R. J. Sternberg, *The Cambridge Handbook of Creativity* (pp. 3-19). Cambridge University Press.

Russell, S., & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach* (3 ed.). Prentice Hall.

Sachs, C. (2012). *The history of musical instruments.* Courier Corporation.

Sánchez, C., Moreno, F., Albarracin, D., Fernandez, J. D., & Vico, F. J. (2013). Melomics: A Case-Study of AI in Spain. *AI Magazine, 34*(3), 99-103.

Sánchez-Quintana, C. A. (2014). Aportaciones y Aplicaciones de Disciplinas Bioinspiradas a la Creatividad Computacional. Doctoral dissertation. Málaga: Universidad de Málaga.

Schaffer, J. W., & McGee, D. (1997). *Knowledge-based programming for music research* (Vol. 13). AR Editions, Inc.

Schedl, M., Gómez, E., & Urbano, J. (2014). *Music Information Retrieval: Recent Developments and Applications.* now Publishers.

Serrà, J., Corral, Á., Boguñá, M., Haro, M., & Arcos, J. L. (2012). Measuring the Evolution of Contemporary Western Popular Music. *Scientific reports, 2*.

Service, T. (2012, July 1). Iamus's Hello World! – review. *The Guardian*.

Shea, K. C., & Fenves, S. J. (1997). A shape annealing approach to optimal truss design with dynamic grouping of members. *Journal of Mechanical Design, 119*(3), 388-394.

Smith, S. (2013, January 3). Iamus: Is this the 21st century's answer to Mozart? *BBC*.

Spector, L., & Alpern, A. (1994). Criticism, culture, and the automatic generation of artworks. *AAAI*, (pp. 3-8).

Stanley, K. O., & Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life, 9*(2), 93-130.

Steels, L. (1979). Reasoning modeled as a society of communicating experts. M. A. Thesis. Massachusetts Institute of Technology, Cambridge.

Steinbeis, N., & Koelsch, S. (2009). Understanding the intentions behind man-made products elicits neural activity in areas dedicated to mental state attribution. *Cerebral Cortex, 19*(3), 619-623.

Steinberg, K. (1999). Steinberg Virtual Studio Technology (VST) Plug-in Specification 2.0 Software Development Kit. Hamburg: Steinberg Soft- und Hardware GMBH.

Sternberg, R. J., & O'Hara, L. A. (1999). 13 Creativity and Intelligence. In R. J. Sternberg (Ed.), *Handbook of creativity* (pp. 251-272). Cambridge, MA: Cambridge University Press.

Stiny, G. (1980). Introduction to shape and shape grammars. *Environment and planning B, 7*(3), 343-351.

Stranges, S., Tigbe, W., Gómez-Olivé, F. X., Thorogood, M., & Kandala, M. B. (2012). Sleep problems: an emerging global epidemic? *Sleep, 35*(8), 1173-81.

Tatarkiewicz, W. (1980). Creativity: History of the Concept. In W. Tatarkiewicz, *A History of Six Ideas an Essay in Aesthetic* (pp. 244-265).

The sounds of Nature Collection (Jungle River, Jungle birdsong and Showers). (2010, 11 5). Retrieved 2015, from Internet-Archive: https://archive.org/details/Sounds_of_Nature_Collection

Thomas, M. T. (1985). Vivace: A rule based AI system for composition. *Proceedings of the International Computer Music Conference*, (pp. 267-274).

Thomas, M. T., Chatterjee, S., & Maimone, M. W. (1989). Cantabile: A rule-based system for composing melody. *Proceedings of the International Computer Music Conference.*

Thywissen, K. (1999). GeNotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition. *Organised Sound, 4*(2), 127-133.

Tokui, N., & Iba, H. (2000). Music composition with interactive evolutionary computation. *Proceedings of the 3rd international conference on generative art*, *17*, pp. 215-226.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 433-460.

Turner, S. R. (2014). *The creative process: A computer model of storytelling and creativity.* Psychology Press.

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE transactions on Speech and Audio Processing, 10*(5), 293-302.

Ullmann, P. (2009). *Patent No. US20090149699 A1.*

Vico, F. J. (2007). *Selfo: A class of self-organizing connection games.* Technical report ITI 07-7. Dpto. Lenguajes y Ciencias de la Computación (Universidad de Málaga).

Vico, F. J., Sánchez-Quintana, C. A., & Albarracín, D. D. (2011). MELOMICS Contributions of computer science and biology to receptive music therapy. Evidence for Music Therapy Practice, Research & Education. *Selected Readings & Proceedings of the VIII European Music Therapy Congress* (págs. 521-530). Cádiz, Spain: Granada: Grupo Editorial Universitario.

Wallas, G. (1926). *The art of thought.* New York, Harcourt, Brace and Company.

Wang, A. (2013). An Industrial Strength Audio Search Algorithm. *ISMIR*, (pp. 7-13).

Williams, D., Kirke, A., Miranda, E. R., Roesch, E., Daly, I., & Nasuto, S. (2015). Investigating affect in algorithmic composition systems. *Psychology of Music, 43*(6), 831-854.

Wilson, A. J. (2009). A symbolic sonification of L-systems. *Proceedings of the International Computer Music Conference*, (pp. 203-206).

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques.* New York: Morgan Kaufman.

Wooller, R. W., & Brown, A. R. (2005). Investigating morphing algorithms for generative music. *Third Iteration: Third International Conference on Generative Systems in the Electronic Arts.* Melbourne, Australia.

Yannakakis, G. N. (2012). Game AI revisited. *Proceedings of the 9th conference on Computing Frontiers* (pp. 285-292). ACM.

This research work could not have been completed as it is without an intensive use of the Wikipedia®.[64] This multilingual, free-access and free-content internet encyclopedia has become an ideal and essential tool for most of us, for accessing knowledge of any kind, in any field. During my research and particularly along the development of this thesis, Wikipedia can be considered as the first bibliographic source, which is then able to lead to the appropriate concrete lines and works. I am completely grateful to all the skilled and enlightened devoted contributors, who make possible these contents to be instantaneously updated, fixed and always reachable.

[64] https://www.wikipedia.org/